Robust Surrogate Models for Uncertainty Quantification and Nuclear Engineering Applications

Uchenna Oparaji

Institute for Risk and Uncertainty, School of Engineering

University of Liverpool

A thesis submitted for the degree of

Doctor of Philosophy (Dual-PhD)

November 2017

國 立 清 華 大 學

博 士 論 文

不確定度量化分析的代理人模型及其在核工的應用

Robust Surrogate Models for Uncertainty Quantification and Nuclear Engineering Applications

系別：核子工程與科學研究所

學號：104013891

研 究 生： Bright Uchenna Oparaji

指導教授： Dr. Edoardo Patelli

許榮鈞：Prof. Rong-Jiun Sheu

中 華 民 國 一○七 年 二 月

I dedicate my PhD thesis to my parents, Mr and Mrs M A Oparaji.
Their support was to different extents crucial for the completion of my thesis.

Acknowledgements

I would like to express my sincere gratitude towards my supervisors Dr Edoardo Patelli and Professor Rong-Jiung Sheu for giving me the opportunity to take part in the first dual PhD program between the University of Liverpool and National Tsing-Hua University Taiwan. This has been an amazing lifetime experience for me as I have had the opportunity to work in different multi-cultural research groups, broaden my network, and expand the visibility of my research. In particular, Edoardo Patellis guidance and support have been crucial to make my work visible on a global scale and for developing key academic partnership by giving me the opportunity to attend a variety of international conferences and workshops. Also, his personality and sense of humour made my research unique and entertaining. I am very grateful to him. Professor Rong-Jiun Sheu has been more than just a supervisor, he has given me fraternal support, constant guidance, and has taken me on a fantastic journey upon my arrival in Taiwan, integrating me effortlessly into his research group. I am very grateful to him for having taken me on board. I also acknowledge Matteo Broggi's help, as his understanding of computational analysis and his programming abilities considerably helped me in gaining familiarity with Matlab and OpenCossan. I would like to thank the National Nuclear Laboratory (NNL) for providing a realistic case study to demonstrate the applicability of the proposed approaches developed in this thesis. I am also very grateful to my colleagues at the Institute of Risk and Uncertainty and friends, who proof-read and significantly improved the presentation of this thesis. I am also grateful to Karen who helped me translate my abstract into traditional mandarin language. Special thanks to my family, who have always motivated me to push harder during difficult times. Finally, I give God the glory for giving me the wisdom, knowledge and understanding to complete the research work presented in this thesis.

Abstract

Nowadays, mathematical models are a popular tool used to design systems of increasing complexity and to determine their performance. These models aim at reproducing the physical process at hand by solving complex mathematical equations. Such models exploit the available computational resources to solve these complex mathematical equations, meaning that a single run of the model can take up to hours or even days to compute a quantity of interest. Typically, the parameters of these models are uncertain, but are inferred from data, modelled probabilistically and propagated through the model. However propagating parameter uncertainty through mathematical models becomes expensive for complex and highly reliable models/systems. Consequently, surrogate models which are easy to evaluate functions are used in place of these expensive models to speed up uncertainty propagation. On the other hand, the use of surrogate models introduces additional model uncertainty that originates from sources such as: (1) variability in training data set, (2) random model parameters, and (3) model structure, thus underestimating or overestimating the quantity of interest sorted out for. Therefore, in this thesis, several frameworks are proposed to quantify the sources of model uncertainties. In the first framework proposed, the model uncertainty that originates from the variability in training data set is quantified. This kind of uncertainty arises from the sampling algorithm used to sample from the input domain of the training data. The sampling algorithm usually tend to sample frequently from high probability regions of the input space, ignoring the low probability regions. This leads to an omission of important training data point for training the surrogate model, thus reducing the generalization properties of the model. Thus, the underlying principle of this approach is to generate random samples with replacement from the original training data set, train an ensemble of surrogate models with this bootstrapped data to make an inference of the about the population where the data set was obtained from. Subsequently, to show the effectiveness of this

framework, feed forward artificial neural network (FF-ANN) is used as a surrogate to test the framework on various analytical functions and the uncertainty quantification of an expensive radioactive waste management model (Site Ion eXchange Plant (SIXEP)) situated at Sellafield nuclear waste site, in terms of computing a robust confidence intervals that quantifies the aforementioned source of uncertainty in the quantities of interest. In the second framework, an approach that quantifies model uncertainty resulting from the random fluctuation of model parameters is proposed. Specifically, when a unique surrogate model is trained repeatedly with the same training data, different models are resulted. Usually, it is of common practice to select the best model out if the set based on some performance metric, and discard the rest. However, there are several issues with this method, the most important being the waste of computational resource. Furthermore, the performance metric evaluated for each model is biased because of the random noise component present within the test data. Hence, a model that performed well during this test might perform bad with a different data set. Thus, there is uncertainty in selecting the best performing model due to the random fluctuating model parameter. To quantify the model uncertainties here, the approach proposed in this framework combines an ensemble of identical surrogate model based on the unification of Bayesian statistics and model averaging technique into a single framework. Like the previous approach in terms of applicability, different analytical examples are tested. Furthermore, the SIXEP model and the fault diagnostics of a nuclear power plant is analysed with this approach adopting FF-ANNs. In the third framework, an approach that quantifies model uncertainty arising from model structure is proposed. This present framework extends the previous framework by including of a multi-objective optimization problem that is aimed at locating global optimal model structures within the given design space. Again, the applicability of the proposed approach is tested on several analytical examples and further tested on the fault diagnostics of a nuclear power plant. The final framework proposed in this thesis combines all the approaches proposed in this thesis into a single unified framework. The advantage of this framework compared to others is that fact that all the aforementioned sources of uncertainties are taken into account. In terms of applicability, different case studies from the field of nuclear engineering are tested.

The variety of examples shows the flexibility and versatility of the proposed framework. Hence, the proposed framework is of importance for the engineering practice when any type of surrogate model is adopted.

Abstract

當今，數學模型被廣泛的應用在設計越來越複雜的系統并判定這些系統的性能。這些模型旨在通過求解複雜的數學方程來重現物理過程。它們利用可用的計算資源來解出複雜的數學方程，往往在模型的進行運算可能花費數小時甚至數天的時間來計算出所需的數量。主要於模型的參數通常是不確定的，但它們是由數據中被推導出來，透過模型進行概率性數值遞迴分析計算。然而，對於複雜且高度可靠的模型和系統來說，通過數學模型來傳播參數不確定性的代價非常高昂。易於加速求解的過程，從而評估函數的替代模型被用來取代原本的數學模型。另一方面，替代模型的使用引入了額外的模型不確定性，其來源包括：（1）訓練數據集的變異性，（2）隨機模型參數，（3）模型結構，導致錯誤估計了所需的數量。因而，本論文提出了幾個框架來量化模型不確定性的來源。

在提出的第一個框架中，模型的不確定性來源於訓練數據集的可變性，即訓練數據輸入域的採樣算法。採樣算法通常傾向於輸入空間的高概率區域採樣，而忽略了低概率區域。這導致了培訓替代模型的重要培訓數據點的遺漏，從而減少了模型的泛化屬性。因此，這種方法的基本原理是從原始的訓練數據集中生成隨機的樣本，用這種引導數據來訓練一組代理模型，從而推斷出從哪裡獲得數據集的總體。隨後，從而展示該框架的有效性，依據計算一個穩健的置信區間，量化上述數量的不確定性的來源，前饋人工神經網絡（FF-ANN）被作爲代替模型用來測試了該框架的各種分析性能及其對一個坐落在塞拉菲爾德核廢料站的昂貴的放射性廢物管理的不確定性量化模型（Site Ion eXchange Plant (SIXEP)）的量化。

第二個框架提出了一種通過模型參數的隨機波動來量化不確定性的方法。具體地說，當一個特定的代理模型被相同的訓練數據反復精化時，從而產生不同的模型。通常情況下，根據某些性能指標選擇最好的模型并捨棄其餘部份是一種常見的做法。但這種做法有幾個問題，最明顯的是會浪費計算資源。此外，由於測試數據中存在的隨機不可預期的變量，每個模型的性能指標都有偏差。因此在一個測試中表現良好的模型可能對另外的數據集產生非所預期的結果。因而，基於隨機波動模型參數而確定最佳表現模型存在不確定性。了量化模型的不確定性，這個框架中提出的方法結合了基於貝葉斯統計和模型平均技術的整合模型。與前述的方法一樣，不同的分析例子的適用性也被測試過。此外，本論文採用該前饋人工神經網絡對替代模型SIXEP和某核電站的故障診斷進行了分析。

第三個框架提出了一種量化源自模型結構的模型不確定性的方法。通過把一個旨在于指定空間內定位全局最優化模型結構的多目標優化問題考慮在內，該框架對先前的框架進行了擴展。再次，對該方法的適用性進行了幾個分析實例的測試，並對核電站的故障診斷進行

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Personally, I see research like riding a bicycle, the more I push the pedal, the more I discover new ideas."

1.1 Context

In state-of-the-art engineering practices, mathematical models are used to represent physical processes or engineering processes systems. There are vast applications of these models in various disiplnes such as the modelling of nuclear waste management systems (see [1]), the modelling of buildings with finite element models (FEM) (see [3]) etc. The mathematical model aims at replicating the behaviour of the physical system for a given set of input parameters. For example, the nuclear waste management model may predict the concentration levels of a particular radionuclide for a given period of time. The set of the model input parameters may include the physical and mechanical properties of the radioactive waste plant. On the other hand, due to the diversity of the applications and the kind of the analyses to be performed, the complexity of the mathematical model may range from simple analytical functions to complex mathematical equations (e.g. partial differential equations). Often, complex mathematical equations have no unique solution method, but rely on solvers such as finite element or finite difference schemes. Nowadays, a large number of commercial and non commercial software packages have been developed intending to solve these complex mathematical models. Despite the advances in these software packages, they all tend to be simplifications of reality. When having identified a suitable mathematical model, different sources of inaccuracies may occur. As the mathematical model represents a simplification of reality, it usually contain model error. Additionally, these numerical models introduce numerical discretization errors. Most importantly,

the parameters of a mathematical model might not be known precisely. The different types of uncertainties might or might not be present in a given problem setting. Hence, it is important to quantify the uncertainties in order to be able to interpret the results realistically. The main source of uncertainty identified in this literature is Aleatory uncertainty. Aleatory uncertainty (from Latin alea, rolling of dice), also called variability, refers to the intrinsic randomness of a natural phenomenon, which cannot be reduced by acquiring more data, and is described using probabilistic models. In the multidisciplinary design of complex critical systems, ignoring the effect of uncertainty is unacceptable, as it may lead to catastrophic consequences. Additionally, complex critical systems involve high-consequence decisions often made on the basis of quantitative data that is very scarce or prohibitively expensive to collect. Despite the availability of detailed high fidelity physical models, engineering practitioners still need to make clear decisions based on available information. Thus, they must be able to trust the methodology adopted to analyse the system, in order to quantify the risk with the available level of information, and so avoid wrong decisions due to artificial restrictions introduced at the modelling stage. Comprehensive modelling of uncertainty that accounts for inherent variability provide insight into engineering problems, enabling robust decisions to be made. Risk is conventionally understood as the product between the failure probability of the system and the expected loss (or consequence) caused by the system failure. While the expected loss is quantified in monetary units, the failure probability is calculated, by means of reliability methods, within a rigorous mathematical framework. Usually, this requires the specification of precise distribution models (of probability), including dependencies for the input variables. The uncertainty management requires the uncertainty to be propagated, quantified and the risk to be assessed and subsequently minimised. Likewise, a system performance can only be improved if the parameters that affect the performance significantly are identified and focused on. Sensitivity analysis is used to achieve this by identifying and ranking the contributions of parameters of the system to the variability in the output quantity of interest. Most often, the variance based method to sensitivity analysis [4] is adopted when assessing the contributions of the state variables. This method is a class of simulation approaches that is used to decomposes the output variance into parts that can be attributed to the inputs and interactions between them. Usually, engineering models are often quite detailed and may require several hours for a single deterministic analysis. On the other hand, to quantify the effect of uncertainty on these systems, models need to be evaluated several times with different combinations of the input parameters. Obviously, this

leads to a computational problem, as a repeated number of model calls is required for robust analyses. Therefore, the development of fast accurate surrogate models is key to make the uncertainty quantification ever closer to the community of engineering practitioners.

1.2 General Framework for Uncertainty Quantification in this Thesis

Uncertainty Quantification (UQ) is a general term to summarize the various task and challenges described previously. A summary of the main elements of UQ framework is listed in the following steps:

Step 1: The computational model representing the physical system or process may consist of an analytical function in the simplest case. However, the computational model may also consist of an entire work-flow containing different pieces of software coupled together or even physical experiments. In general, the computational model map a set of input parameters to the quantity of interest that is used in decision making.

Step 2: The characterisation of input parameter uncertainties aims at identifying the type of uncertainty and modelling them adequately. A variety of modelling choices for different types of uncertainties are available, which includes constants (i.e. no uncertainty), probability distributions, intervals, and imprecise probabilities. The suitability of an uncertainty model depend on various factors such as the availability of information of the input parameters, general knowledge of the system being analysed, as well as the purpose of the analysis. For the purpose of this thesis, probability distributions which comes under the framework of probability theory will be used to model the parameter uncertainties.

Step 3: Uncertainty propagation analyses how the uncertainty in the input parameters is transformed through the computational model towards the quantity of interest in the output of the model. When the input is modelled by uncertain parameters, the quantity of interest is likely to be uncertain too. Hence, this step analyses different statistics of interest, depending on the problem at hand. In this thesis, the statistics of interest include the failure probabilities, and sensitivity indices.

1.3 Problem Statement

In the general framework for UQ, there are various challenges. Starting from Step 1, consider the computational model that represents the physical system being analysed (i.e. FEM model representing a building). The model may be complex (i.e. large number of parameters) and expensive to evaluate such that a single run of the computational model takes minutes, hours or even days to converge to a solution. The computational model is usually given as a black box such that only the input parameters and the output quantity of interest are observable. Thereafter, in Step 2, probability theory is used to characterize and model the aleatory uncertainty. Nevertheless, when the computational model at hand is expensive to evaluate due to complexity of the model, the total computational cost is dominated by increased wall-clock time. Consequently, surrogate-modelling (also called response surface and meta models) techniques are widely established in probabilistic settings. Surrogate models approximate the expensive computational model by a simple and inexpensive to evaluate function. Subsequently, these surrogate models can be used for uncertainty propagation analyses such as reliability analysis and sensitivity analysis, allowing for high number of model evaluations for reduced wall-clock time. Contrarily, the use of a surrogate model for this kind of analysis can introduce some additional uncertainty into the quantity of interest sorted out. Clearly, there is a need to quantify the additional uncertainties introduced by the surrogate model in order to ensure robust and reliable results, in particular, when the results obtained from the surrogate model is used for decision making.

1.4 Objectives of the Thesis

Considering the problem statement, this thesis is focused on the following objectives:

- To develop intuitive frameworks to quantify the surrogate model uncertainties.
- To utilize state-of-art surrogate modelling techniques to conduct uncertainty quantification analysis in the presence of aleatory uncertainties.
- To apply the proposed frameworks to a variety of classical uncertainty quantification analyses problems, such as structural reliability analysis and sensitivity analysis.
- To show the relevance of the proposed frameworks for other realistic nuclear engineering problems.

1.5 Original Contributions

This thesis provides numerical techniques that contributes to the field of surrogate modelling for Uncertainty Quantification and Regression Analysis. Although the techniques proposed in this thesis was originally meant for artificial neural networks, it can easily be adopted to different classes of surrogate models. Particularly, the techniques developed in this thesis are used for quantifying the model uncertainties introduced when using a surrogate model for Regression Analysis, and Uncertainty Quantification. the sources of uncertainties considered in this thesis originates from:

- Variability in training data set.
- Random model parameters.
- Structure of the model

To the author knowledge, there is no generalized numerical framework that tackles all theses sources of uncertainties affecting the performance of a surrogate model used in the aforementioned tasks. Thus, there is a need to develop such techniques, particularly, when surrogate models are used for safety critical applications.

1.6 Numerical Implementation

The quantification of surrogate model uncertainty require the availability of flexible numerical tools. For these reasons, the proposed numerical frameworks have been developed and integrated into **OpenCossan** (see [5]). **OpenCossan** is a collection of open source algorithms, methods and tools released under the LGPL licence, and under continuous development at the Institute for Risk and Uncertainty, University of Liverpool, UK. The source code of the software is available upon request. **OpenCossan** is also the computational core of the general purpose software, called COSSAN-X, which was originally developed by the research group of Prof. G.I. Schuller at the University of Innsbruck, Austria [6]. The term general purpose software implies that a wide range of engineering and scientific problems can be treated with a single software. The computational core of the software is developed in MATLAB object-oriented environment, which includes several predefined solution sequences to solve problems from different fields. The framework is organized in classes, consisting of properties and methods together with their interactions and interfaces. Thanks to the modular nature of **OpenCossan**, it is possible to define specialized solution sequences for robust surrogate modelling and parallel computing strategies to reduce

the overall cost of implementing the proposed framework to the practical engineering problems in this thesis.

1.7 Outline of Thesis

The organisation of chapters in this thesis follows the simple structure shown in Figure 1.1. In Chapter 2, the theoretical background of uncertainty modelling, propagation and quantification is introduced. Thereafter, Chapter 3 gives a brief overview of the state-of-the-art in surrogate modelling, with emphasis paid to Artificial Neural Networks (ANNs). In Chapter 4, an adaptive bootstrap technique used to quantify the model uncertainties of ANN originating from variability in training data. In Chapter 5, a framework used to deal with model uncertainties originating from random parameters of the surrogate model is introduced. Subsequently, Chapter 6 extends the framework introduced in Chapter 5 by including another framework that considers the model uncertainties originating from the structure of the surrogate model. The applicability of the proposed frameworks in Chapter 4 and 5 are tested on real case study in Chapter 7. Furthermore, in Chapter 8, the framework developed in Chapter 6 is tested on another real case study concerning fault diagnostic of a nuclear power plant. In addition, the approach developed in Chapter 4 is combined with the approach in Chapter 6 and applied to the same case study. In Chapter 9, the framework proposed in chapter 4 is applied to real Nuclear engineering problem faced at high-energy neutron facilities. Lastly, some conclusions and recommendations are provided in Chapter 10, summarising the presented work and indicating directions for potential future developments.

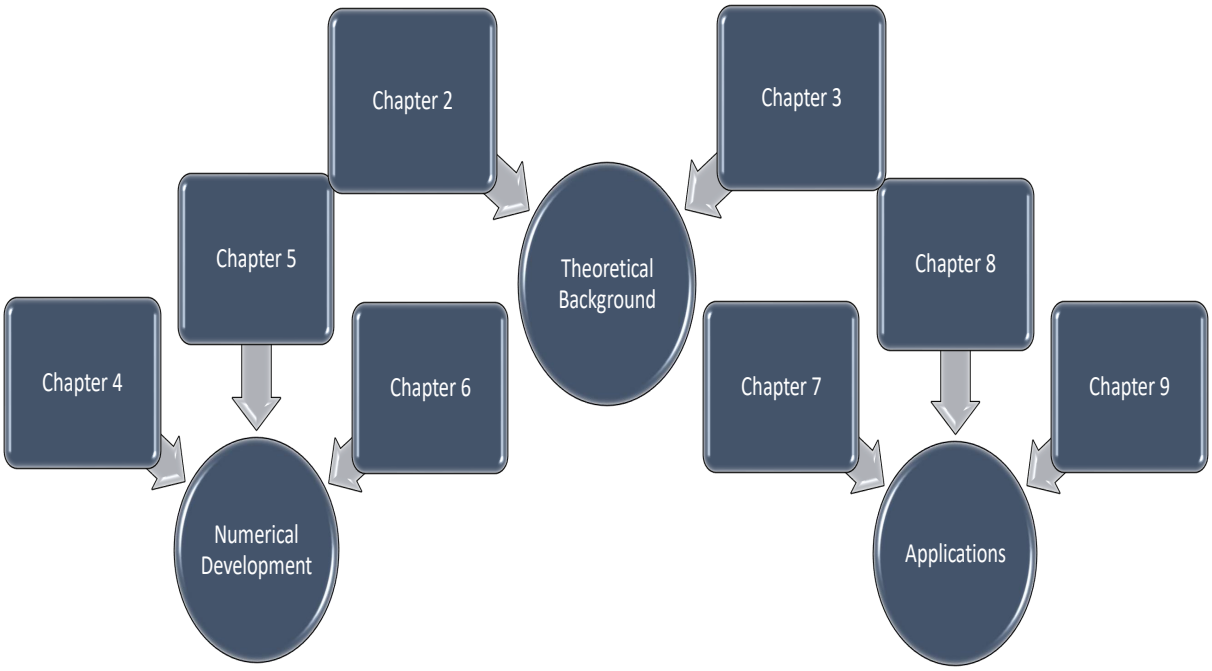


Figure 1.1: Organisation of Chapters in Thesis

Chapter 2

Modelling and Quantification of Parameter Uncertainties

Parameter uncertainty can be modelled by a variety of concepts. In this chapter, random variable which is the selected choice to model aleatory uncertainty in this thesis is discussed. In addition, uncertainty propagation and quantification techniques such as reliability and sensitivity analyses are discussed.

2.1 Modelling Aleatory Uncertainty

2.1.1 Probability Theory

In a probability space $(\Omega, \mathcal{F}, \mathbb{P})$, Ω denotes an event space (also called sample space, universal set, or outcome space) equipped with the σ-algebra $\mathcal{F}$ and a probability measure $\mathbb{P} \in [0, 1]$. Furthermore, for an event $\mathcal{E} \in \Omega$, and its complementary event $\mathcal{E}^c$, by definition, their union is given as $\mathcal{E} \cup \mathcal{E}^c = \Omega$ and their intersection gives a zero event $\mathcal{E} \cap \mathcal{E}^c = \emptyset$. Hence, the probability of $\mathcal{E}$ and $\mathcal{E}^c$ add up to one: $\mathbb{P}(\mathcal{E}) + \mathbb{P}(\mathcal{E}^c) = 1$. The respective probability of the empty set $\emptyset$ and the complete set are given as $\mathbb{P}(\emptyset) = 0$ and $\mathbb{P}(\Omega) = 1$. With this condition, a random variable X is defined by the mapping $X(\omega) : \Omega \mapsto \mathcal{D}_X \subset \mathbb{R}$, where $\omega \in \Omega$ is an elementary event and $\mathcal{D}_X$ is the support domain of X. A realization of the random variable X is denoted by the corresponding lower case letter x. A random variable X is normally characterized by a cumulative distribution function (CDF) F_X which assigns a probability to the event $\{X \leq x\}$, i.e. $F_X(x) = \mathbb{P}(X \leq x)$. From this definition, any CDF that is monotonically non-decreasing, tends to zero for low values of x, and tends to one for large values of x. For continuous random variables, the first derivative of the CDF is the probability density function (PDF) and is denoted by $f_X(x) = dF_X(x)/dx$.

The PDF describes the likelihood of X being in the neighbourhood of x. Due to the monotonicity property of CDFs, $f_X(x) \geq 0$ for all $x \in \mathcal{D}_X$.

2.1.1.1 Data to Cumulative Distribution Function

Empirical Cumulative Distribution Function

Consider a set of sample realizations $\chi = \{\chi^{(1)}, ..., \chi^{(N)}\}$ of a random variable X, whose probability distribution is not known. To describe χ properly, the data χ is used to deduce a probability distribution. A basic method of statistics is then to compute the empirical CDF, which is defined as:

$$F_X^{emp} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_{x \geq \chi^{(i)}}(x), \tag{2.1}$$

where $\mathbb{I}$ is the indicator function, which indicates $\mathbb{I} = 1$ for a true subscript statement and $\mathbb{I} = 0$ otherwise. As the number of samples N are limited, the empirical CDF is a stair-shaped curve with constant CDF values between samples and steps at $x = \chi^{(i)}, i = 1, ..., N$. Assuming that the probability distribution X may be continuous, the empirical CDF provides a poor but simple estimate of F_X. Thus, more sophisticated methods, such as the method of moments or the maximum likelihood method, are usually adopted for practical purpose.

Method of Moments

Here, a distribution family $F_X(x|\theta)$ is considered, where θ denotes a vector of parameters that defines the shape of the CDF. In addition, the number of parameters is denoted as $n_\theta = |\theta|$. Commonly used distributions in literature and practice, have $n_\theta = 2$. Next, the method of moments determines the optimal distribution by matching the first moments of $F_X(x|\theta)$ with the sample-based estimations of the first moments based on χ, by denoting the mean value and variance of X by $\mu(\theta)$ and $\sigma^2(\theta)$, which depend on the yet-to-be-determined parameters θ. Further, the sample mean and variance is given by $\mathbb{E}[\chi]$ and $Var[\chi]$:

$$\mathbb{E}[\chi] = \frac{1}{N} \sum_{i=1}^{N} \chi^{(i)}, \tag{2.2}$$

and,

$$Var[\chi] = \frac{1}{N-1} \sum_{i=1}^{N} (\chi^{(i)} - \mathbb{E}[\chi])^2. \tag{2.3}$$

The parameters θ are obtained by solving: $\mu(\theta) = \mathbb{E}[\chi]$, $\sigma^2(\theta) = Var[\chi]$. The method of moments relies on the knowledge of the underlying distribution family, which is

an additional assumption compared to the empirical CDF. However, this assumption avoids the stair-shaped CDF curves of the empirical CDF, thus providing a smooth CDF curve.

Maximum Likelihood Method

Analogous to the method of moments, the maximum likelihood method requires the knowledge of a distribution family $F_X(x|\theta)$ with unknown distribution parameters θ. Thereafter, the likelihood of observing χ depending on the parameter value θ is computed by:

$$\mathcal{L}(\theta|\chi) = \prod_{i=1}^{N} f_X(\chi^{(i)}|\theta), \tag{2.4}$$

where $f_X(x|\theta)$ is the PDF conditional on θ. The optimal parameter values are deduced by maximizing the likelihood function:

$$\theta^* = arg\ \max_{\theta}\ \mathcal{L}(\theta|\chi). \tag{2.5}$$

The practicability of the method depends on the initial assumption on the distribution family. In fact, any distribution family can be fitted using this method.

2.2 Quantification of Parameter Uncertainties

Once the parameter uncertainty has been modelled with a PDF, propagation of the parameter uncertainties through the model to the quantify of interest is required for an adequate quantification of the uncertainties. To quantify the performance of complex critical systems in the presence of the parameters uncertainties, reliability analysis is carried out. Similarly, a system performance can only be improved if the parameters that affect the performance significantly are identified and focused on. Sensitivity analysis is used to achieve this by identifying and ranking the contributions of each state variable of the system to the variability in the performance. Henceforth, the following sections provides a concise discussion on the theory of reliability and sensitivity analysis using the simulation approach.

2.2.1 Reliability Analysis of Systems

Reliability methods is a powerful technique used in various engineering disciplines to quantify the performance of the system under investigation in the presence of parameter uncertainties. The aim of reliability methods is producing a design that meets

some predefined performance objectives. For example, during the lifetime of a structure, a series of conditions that depends on external actions affects the performance of the structure. Usually, the external actions are random processes in space and time. Hence, the external actions affect the structural responses. The structural responses are referred to as demands, while the thresholds that the responses are not allowed to exceed are referred to as capacities. Further, the space of all demands and capacities is referred to as state space which is used to form the limit state surface of the structure. The intersection and union of this limit state surfaces define two mutually exclusive domains: the failure domain, χ_F , and the survival domain, χ_S. A random state, x, of the system, which includes the demands and capacities, can be represented as a point in the state space, where the structure is in a "safe" state if the point is strictly contained in the safe domain, $x \in \chi_S$, whereas it is in a "failed" state if contained in the failure domain, $x \in \chi_F$. The reliability, p_R, is formulated as the probability of a random state to be in the safe domain, whereas, the probability of failure, p_F is the probability of a random state to be in the failure domain. Hence, $p_F + p_R = 1$. In the state space, some reliability metrics can be defined to assess to what extent the structure can be considered safe. For instance, it is common to refer to the distance between a random state and the limit state surface as the safety margin. The failure probability p_F is defined as:

$$p_F = \int \cdots \int_{\chi_F} f_X(x_1, \ldots, x_d)\, dx_1 \ldots dx_d, \tag{2.6}$$

where, f_X denotes the joint density function, and d is the number of model input parameters. In general, the integral of Eq. (2.6) is difficult to calculate analytically, due to the multi-dimensional nature of the integral. Thus, its estimation can be analytically intractable, especially for complex systems with a large number of parameters.

2.2.1.1 Estimation of the Failure Probability by means of Simulation

Consequently, various methods have been proposed to compute p_F in recent and past literature. The first attempts were oriented towards using analytical methods [7]. Afterwards, numerical methods based on the calculation of the performance function Hessian [8] or asymptotic approximation [9] were increasingly used. Conversely, these numerical methods becomes inefficient as the number of parameters increases, thus, the error in the estimate of p_F increases for complex models. Hence, simulation methods based on Monte Carlo (MC) simulation were proposed [10, 11]. In particular,

MC simulation is independent of the system complexity and is the most flexible method to estimate p_F [6]. Fundamentally, MC simulation is performed by means of sampling a large number of realisations from the joint distribution, x, from given probability distributions, thereafter counting the number of "safe" samples over the total number of samples. Specifically, an indicator function, $\mathbb{I}_F$, is used to label the states, such that it indicates zero if "safe" and one if "unsafe". Thus, the failure probability is estimated as:

$$p_F = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \mathbb{I}_F[x \in \chi_F] f_X(x_1, ..., x_d) dx_1, ..., dx_d. \tag{2.7}$$

If the samples generated, $x^{\{s\}}, s = 1, ..., N_s$, follows the pattern of the distribution f_X, Eq. (2.7) can be approximated by:

$$\hat{p}_F = \frac{1}{N_s} \sum_{s=1}^{N_s} \mathbb{I}_F[x^{\{s\}}] \in \chi_F, \tag{2.8}$$

by generating a large number of samples, N_s, which provides an unbiased estimation of the failure probability. The accuracy of the estimate of Eq. (2.8) solely depends on the number of samples generated and can be assessed computing the coefficient of variation, CoV , of the estimator, $\hat{p}_F$. The CoV of the failure probability estimator is defined as:

$$CoV[\hat{p}_F] = \sqrt{\frac{1 - p_F}{p_F \times N_s}}. \tag{2.9}$$

This simply means that a sample size $N_s > 1/p_F$ is required for $CoV[\hat{p}_F] < 1$ and acceptable values, where $CoV[\hat{p}_F] < 0.3$, can be obtained for $N_s = 10/p_F$. Thus, a large number of model evaluations are required by the method in order to assess the indicator function, $\mathbb{I}_F$. The large number of model evaluations can be infeasible in realistic engineering practices, as the computational time for a single model evaluation can be long. Subsequently, the limitations of direct Monte Carlo can be mitigated by resorting to Advanced Sampling methods, such as Importance Sampling (IS) [12], Line Sampling (LS) [3, 13], Subset Simulation (SS) [14, 15], etc. Importantly, each of these simulation method carries their own special performance feature to target different classes of problems. For instance, LS is specially suited to estimate small failure probabilities in high dimensional spaces, provided that the limit state surface displays a single failure mode. However, the use of advanced sampling technique comes with several limitations that could affect the accuracy of the targeted failure probability p_F. For example, in order to accurately compute p_F using the IS technique, the analyst

should know what region of the space to sample from, as IS may underestimate the target failure probability if the proposal distribution does not cover the whole failure region. Similarly, the LS technique requires setting up an important direction which is defined as the direction pointing towards the region of interest. However, a priori knowledge about the system failure domain is required before defining the important direction. Contrarily, LS has been found to significantly outperform SS, in particular in the task of estimating very small failure probabilities (i.e., around 10^{-7}) (see [16]).

2.2.2 Sensitivity Analysis of Systems

On the other hand, a system performance can only be improved if the state variables that affect the performance significantly are identified and focused on. Sensitivity Analysis (SA) is used to achieve this by identifying and ranking the contributions of the input parameters to the variability in the output quantity of interest. Different classes of SA methods can be found in the literature [17–19]. Specifically, the methods are divided into two classes, namely local and global SA methods. In local SA, the effect of small variations in the input variables to the quantity of interest are investigated, whereas global SA focuses on the entire variation of the input variables. In this thesis, global SA methods are focused on. Commonly, global SA methods are developed under the framework of the probability theory, i.e. the uncertainty in the input variables is modelled as random variables. Thus, a large number of evaluations of the computational model for different realizations of the input random variable is required. In this thesis, the variance-based sensitivity analysis method often referred to as Sobol' indices is adopted. In the context of probability theory, the variance based method decomposes the variance of the output quantity of interest into fractions which can be attributed to the inputs. From a black box perspective, the model under investigation is viewed as a function $\mathbf{y} = f(\mathbf{x})$, where $\mathbf{x}$ is a vector of d uncertain model inputs $x_1, x_2, ..., x_d$, and y is a chosen univariate model output (note that this approach examines scalar model outputs, however multiple outputs can be analysed by multiple independent sensitivity analyses). Furthermore, it is assumed that the inputs are independently and uniformly distributed within the unit hypercube, i.e. $x_i \in [0, 1]$ for $i = 1, 2, ..., d$. This incurs no loss of generality because any input space can be transformed onto this unit hypercube. Thus, $f(\mathbf{x})$ may be decomposed as:

$$\mathbf{y} = f_0 + \sum_{i=1}^{d} f_i(\mathbf{x}_i) + \sum_{i<j}^{d} f_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \cdots + f_{1,2,...,d}(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_d), \tag{2.10}$$

where f_0 is a constant and f_i is a function of $\mathbf{x}_i$, f_{ij} a function of $\mathbf{x}_i$ and $\mathbf{x}_j$, etc. A condition of this decomposition is that:

$$\int_0^1 f_{1,2,...,d}(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_d) d\mathbf{x}_d = 0, \tag{2.11}$$

i.e. all the terms in the functional decomposition are orthogonal. This leads to definitions of the terms of the functional decomposition in terms of conditional expected values:

$$f_0 = \mathbf{E}(\mathbf{y}), \tag{2.12}$$

$$f_i(\mathbf{x}_i) = \mathbf{E}(\mathbf{y}|\mathbf{x}_i) - f_0, \tag{2.13}$$

$$f_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{E}(\mathbf{y}|\mathbf{x}_i, \mathbf{x}_j) - f_0 - f_i - f_j. \tag{2.14}$$

It can be seen that f_i is the effect of varying $\mathbf{x}_i$ alone which is the main effect of $\mathbf{x}_i$, and f_{ij} is the effect of varying $\mathbf{x}_i$ and $\mathbf{x}_j$ simultaneously, additional to the effect of their individual variations. This is known as a second-order interaction. Higher-order terms have analogous definitions. Further assuming that the $f(\mathbf{x})$ is square-integrable, the functional decomposition may be squared and integrated to give:

$$\int_0^1 f^2(\mathbf{x}) d\mathbf{x} - f_0^2 = \sum_{i=1}^{d} \sum_{1<...<d}^{d} \int f^2_{1,2,...,d} d\mathbf{x}_i, ..., d\mathbf{x}_d. \tag{2.15}$$

Notice that the left hand side is equal to the variance of $\mathbf{y}$, and the terms of the right hand side are variance terms, now decomposed with respect to sets of the $\mathbf{x}_i$. This finally leads to the decomposition of variance expression:

$$Var(\mathbf{y}) = \sum_{i=1}^{d} V_i + \sum_{i<j}^{d} V_{ij} + \cdots + V_{1,2,\cdots,d}, \tag{2.16}$$

where,

$$V_i = Var_{x_i}(E_{x\ i}(y|x_i)), \tag{2.17}$$

and

$$V_{ij} = Var_{x_{ij}}(E_{x\ ij}(y|x_{ij})). \tag{2.18}$$

A direct variance-based measure of sensitivity S_i, called the first-order sensitivity index, or main effect index is stated as follows:

$$S_i = \frac{V_i}{Var(y)}. \tag{2.19}$$

This is the contribution to the output variance of the main effect of $\mathbf{x}_i$, therefore it measures the effect of varying $\mathbf{x}_i$ alone, but averaged over variations in other input

parameters. It is standardised by the total variance to provide a fractional contribution. Higher-order interaction indices such as S_{ij} can be formed by dividing other terms in the variance decomposition by $Var(\mathbf{y})$. Note that this has the implication that,

$$\sum_{i=1}^{d} S_i + \sum_{i<j}^{d} S_{ij} + \cdots + S_{1,2,\ldots,d} = 1. \tag{2.20}$$

Using the S_i, S_{ij} and higher-order indices given above, one can visualize the significance of each variable in the output variance. However, when the number of variables is large, this requires the evaluation of $2d - 1$ indices, which can be too computationally demanding. For this reason, a measure known as the Total-effect index, T_i, is used. This measures the contribution to the output variance of $\mathbf{x}_i$, including all variance caused by its interactions, of any order, with any other input variables. It is given as:

$$T_i = 1 - \frac{Var_{x\ i}[E_{x\ i}(y|x\ _i)]}{Var(y)}. \tag{2.21}$$

2.2.2.1 Estimating Sobol' Indices by means of Monte Carlo Simulation

Subsequently, to estimate Sobol' indices, MC sampling based methods have been developed in several literatures. For example, in ref [20], a method for estimating the Sobol' first order and interaction indices have been developed. Additionally, in ref [17] a method for estimating the first order and total indices have been developed. Unfortunately, to get robust estimates of sensitivity indices, both methods are costly in terms of the large number of model runs required. Consequently, using quasi-Monte Carlo sequences such as Sobol sequence [21] or Latin hypercube sampling (LHS) [22] for sample generation instead of the classic Monte Carlo (MC) method can sometimes reduce the number of model runs by a factor ten [23]. Furthermore, in the Fourier Amplitude Sensitivity Test (FAST) method [24], a single frequency variable is used to represent a multivariate function in the frequency domain. Thus, the integrals required to compute the Sobol' indices become univariate, resulting in low number of model calls. Consequently, ref [25] have extended the FAST method to compute total Sobol' indices. In addition, ref [26] have coupled FAST method with a Random Balance Design. Recently, ref [27] have analysed and improved these methods. Contrarily, FAST remains expensive, unstable and biased when the number of inputs increases [27]. On the other hand, the advantages of using a MC based method to compute the sensitivity indices is that it provides error made on indices estimates via random repetition or bootstrap methods. Thus, MC based methods are

widely used. Generally speaking, using MC technique to estimate S_i and the T_i, the total number of model calls follows the relationship $N(d+2)$. Hence, such analyses can become intractable when the computational model is expensive-to-evaluate. Then, the use of surrogate models is a popular solution to lower the total computational costs. The following chapter gives a brief overview of surrogate models.

2.3 Chapter Summary

The concepts presented in this chapter gives an overview of how the uncertainty in the model input parameter can be modelled, propagated and quantified within the context of the probabilistic theory. Subsequently, in an attempt to reduce the number of model calls for quantifying the parameter uncertainties, different techniques have been proposed. However, even for the most efficient technique proposed, the number of model calls are still large. Therefore, easy-to-evaluate surrogate models that can be used to replace the expensive models are required to reduce the computational cost. The following chapter gives the current state-of-the-art surrogate modelling techniques available.

Chapter 3

Classical Surrogate Models

This chapter gives an overview of the current state-of-the-art surrogate modelling techniques used for Uncertainty Quantification. In particular, feed-forward Artificial Neural Network and Deep Neural Networks are focused on.

3.1 State of the Art

Uncertainty Quantification (UQ) usually requires a large number of model calls for different input values through a MC simulation procedure. Such approach requires thousands to millions of model calls which is not affordable in many practical cases even with high-performance computing. Subsequently, a viable approach to reduce the computational burden associated to UQ is resorting to easy-to-evaluate surrogate models, also called response surfaces or meta-models. These surrogate models are used in place of the high fidelity expensive model. Although they are fast, they still require the same number of model calls required for UQ, but require a lesser wall-clock time. Furthermore, the construction of surrogate models require running the expensive moddel a predetermined number of times (e.g., 50-100 or more) via design of experiment techniques (see [28]) for specified range of the input variable space. Then, collecting the corresponding values of the output of interest. Thereafter, statistical techniques are used to calibrate the internal parameters of the surrogate model in order to capture the underlying behaviour of the expensive model. The popular types of surrogate models include polynomial chaos expansions (PCE) [29], Kriging (also known as Gaussian process) [30–32], support vector machines (SVM) [33], radial basis function (RBF) [34], reponse surfaces [35], and artificial neural networks (ANNs) [36], which have been extensively investigated in the last decade. Recent development in surrogate modelling such as PC-Krigin have been proposed in ref [37]. This current development combines classical PCE and Kriging. Furthermore,

it has been shown to perform better than either PCE or Kriging (see [37]). Several literatures can be found concerning the application of these aforementioned surrogate models in UQ analysis. For example, in refs [38–40], response surfaces are employed to evaluate the failure probability of structural systems. In [41–44], surrogate models such as ANNs, RBF and SVM are trained to provide local approximations of the failure domain in structural reliability problems. In [45, 46], Kriging models are used to speed up the computation of the global sensitivity indices for a complex hydro-geological model simulating radionuclide transport in groundwater. Finally, in the literature [37, 47, 48] a vast number of surrogate modelling techniques are used for UQ problems. Furthermore, comparing the accuracy of the aforementioned surrogate modelling techniques, Kriging models tend to gives the best predictive performance due to the capabilities of their correction of the trend function, which ensures that the model passes for the value of every sample. Particularly, on small datasets (i.e. small number of samples and low model complexity), Kriging models are accurate because of this well-tuned smoothing property and cheap computational cost. However, for a large multi-dimensional data set (i.e. large number of training samples and complex model) Kriging models tends to under-perform. Thus, a surrogate model that can scale to large datasets and can generalize globally is needed. For this reason, ANNs are been chosen as the primary surrogate models in this thesis. The following sections gives a brief background to ANNs.

3.2 Background to Neural Networks

The human brain is complex machine able to solve complex problems. Although humans have a basic understanding of some of the operations that drive the brain, we are still far from understanding everything there is to know about the brain. Subsequently, in order to understand ANNs, a basic knowledge is required on how the brain functions. The brain is part of the central nervous system and consists of a very large neural network. The neural network in the brain is complicated, but will be simplified for the basic knowledge needed to understand ANNs. The neural network is a network that consist of connected neurons. The center of the neuron is called the nucleus. The nucleus is connected to other nucleus by means of the dendrites and the axon. This connection is called a synaptic connection. The neuron can send electric pulses through its synaptic connections, which is received at the dendrites of other neurons. In particular, when a neuron receives enough electric pulses through its dendrites, it activates and fires a pulse through its axon, which

is then received by other neurons. As a consequence, information can propagate through the neural network. Note that the synaptic connections change throughout the lifetime of a neuron and the amount of incoming pulses needed to activate a neuron also change. As a result, this continuous change allows the neural network to learn efficiently. Generally speaking, the human brain consists of around 10^{11} neurons which are highly interconnected with around 10^{15} connections [49]. These neurons activate in parallel as an effect to internal and external sources. The brain is connected to the rest of the nervous system, which allows it to receive information by means of the five senses and also allows it to control the muscles.

3.2.1 Artificial Neural Network

Currently, it is not possible to make an artificial brain, however it is possible to make simplified artificial neurons and ANN. Importantly, an ANN is not intelligent, but is good for recognizing patterns and approximating non-linear functions. ANN approximate non-linear functions on the basis of learning-by-example, meaning that when it is presented with training examples of the underlying function, it can generalize from these examples in order to approximate the actual function. The task of the ANN is to approximate the output of a function for any valid input, after having seen input-output examples for only a small part of the input space. ANN also have excellent training capabilities which is why they are often used in artificial intelligence research.

3.2.1.1 The Artificial Neuron

The idea of the artificial neuron was proposed by [50], but until [51] proposed the back-propagation algorithm that more focus was paid to ANN research. Currently, the most widely used kind of ANN is the multi-layered feed-forward ANN (FF-ANN), which consists of multiple layers of artificial neurons. The neurons are connected by connections which only go forward in between the layers. The back-propagation algorithm and most other related algorithms trains an ANN by propagating an error value from the output layer and back to the input layer while altering the connections on the way. A multilayer FF-ANN consists of neurons and connections. The neurons are located in layers and the connections go forward between the layers. Each neuron receives multiple inputs from others via connections that have associated weights, analogous to the strength of the synapse. When the weighted sum of inputs exceed the threshold value of the node, it activates and passes the signal through an activation

function and sends it to neighbouring nodes. Figure 3.1 shows a network architecture of a 2 hidden layer feed-forward network.

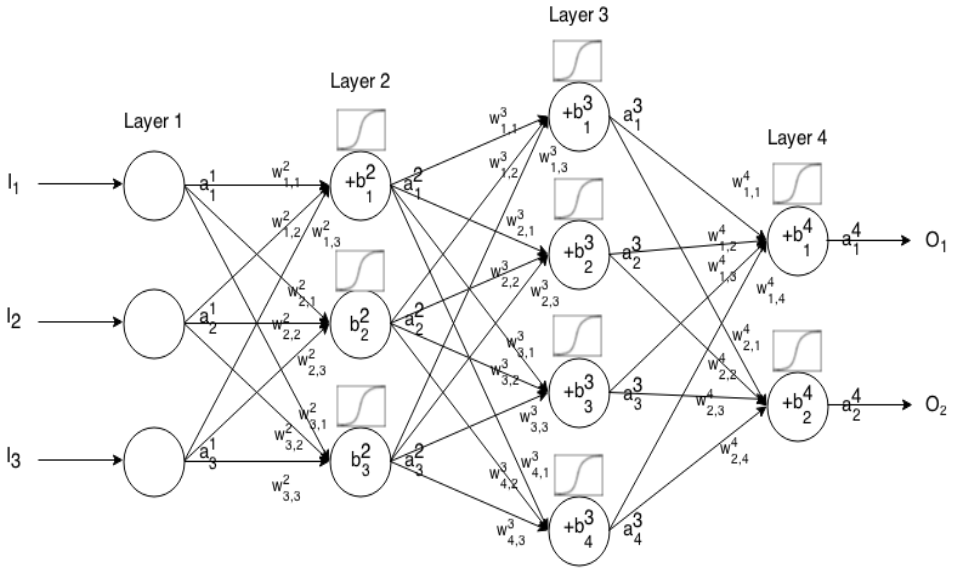


Figure 3.1: Architecture of a 2 Hidden Layer Feed-forward Artificial Neural Network with Neurons

More formally, if we call a_j^i the activation value of the j^{th} neuron in the i^th layer, where a_j^1 is the j^{th} element element in the input vector. Then, the next layer input can the related to the previous layer input via the following relationship:

$$a_j^i = \kappa(\sum_k (w_{jk}^i \times a_k^{i-1}) + b_j^i), \tag{3.1}$$

where κ is the activation function, w_{jk}^i is the weight from the k^{th} neuron in the $(i-1)^{th}$ layer to the j^{th} neuron in the i^{th} layer. b_j^i is the bias of the j^{th} neuron in the i^{th} layer, and a_j^i represents the activation value of the j^{th} neuron in the i^{th} layer. Usually, the activation function κ has many forms. Usually, when using an ANN for regression problems, a non-linear activation function can be used for all neurons except for the neurons in the output layer. On the other hand, in classification problems, non-linear activation function can be used in all layers including the output layer. Furthermore, if the output values from the activation function differ from the target values, the ANN undergoes training to minimize this difference between these values. The dominating training algorithm for training ANN is back-propagation and most other training algorithms are derivations of the standard back-propagation algorithm. There are two fundamentally different ways of training an ANN using the back-propagation algorithm:

- Incremental training the weights in the ANN are altered after each training pattern has been presented to the ANN (sometimes also known as on-line training or training by pattern).
- Batch training the weights in the ANN are only altered after the algorithm has been presented to the entire training set (sometimes also known as training by epoch).

Since training an ANN is simply adjusting the weights to minimize the error function:

$$E = \frac{1}{2}\sum_{i=1}^{N} ||\hat{y} - y||^2. \tag{3.2}$$

E can be minimized using an iterative process of gradient descent, for which the gradient is needed to be calculated:

$$\bigtriangledown E = (\frac{\partial E}{\partial w_{jk}^i}). \tag{3.3}$$

Each weight w_{jk}^i is updated using the increment:

$$\bigtriangledown w_{jk}^i = -\gamma \frac{\partial E}{\partial w_{jk}^i}, \tag{3.4}$$

where γ represents a learning constant, i.e., a proportionality parameter which defines the step length of each iteration in the negative gradient direction. With this extension of the original network the whole learning problem now reduces to the question of calculating the gradient of a network function with respect to its weights. Once we have a method to compute this gradient, we can adjust the network weights iteratively. The minimum of the error function is found where $\bigtriangledown E = 0$. On the other hand, many have viewed minimizing $\bigtriangledown E$ as an optimization problem, which can be solved by techniques used for general optimization problems. These techniques include simulated annealing [52], particle swarm [53], genetic algorithms [54], Levenberg-Marquardt [55] and Bayesian techniques [56]. In addition, an approach which can be used in combination with these training algorithms is ensemble learning [57], which trains a number of networks and uses the average output (often weighted average) as the real output. The individual networks can either be trained using the same training samples, or they can be trained using different subsets of the total training set. Similarly, a technique known as boosting [58] gradually creates new training sets and trains new networks with the training sets. The training sets are created so that they will focus on the areas that the already created networks are

having problems with. These approaches have shown very promising results and can be used to boost the accuracy of almost all of the training algorithms, but it does so at the cost of more computation time. All of these algorithms use global optimization techniques, which means that they require that all of the training data is available at the time of training.

3.2.2 Deep Learning with Artificial Neural Networks

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans (i.e. learn by example). Most deep learning methods use ANN architectures, which is why deep learning models are often referred to as deep neural networks. The term "deep" usually refers to the number of hidden layers in the ANN. Traditional feed-forward ANN architecture only contain 2-3 hidden layers, while deep networks can have as many ranging from hundreds to thousands. The famous types of deep neural network are the Convolution Neural Networks (CNNs) and Recurrent Neural Networks (RNNs).

3.2.2.1 Convolution Neural Networks

In convolution neural networks, each node is connected only to a local region in the input. The local connectivity is achieved by replacing the weighted sums from the neural network with convolutions. In each layer of the convolution neural network, the input is convolved with the weight matrix (also called the filter) to create a feature map. That is to say, the weight matrix slides over the input and computes the dot product between the input and the weight matrix. Note that as opposed to classic feed-forward neural networks, all the values in the output feature map share the same weights. This means that all the nodes in the output detect exactly the same pattern. The local connectivity and shared weights aspect of CNNs reduces the total number of adjustable parameters resulting in more efficient training. The underlying idea behind a convolution neural network is to learn in each layer a weight matrix that will be able to extract the necessary, hidden features from the input. The input to a convolution layer is usually taken to be three-dimensional (i.e. the height, weight and number of channels). In the first layer this input is convolved with a set of M_1 three-dimensional filters applied over all the input channels to create the feature output map. To give a mathematical interpretation of a CNN, we consider now a one-dimensional input $x = (x_t)_{t=0}^{N-1}$ of size N The output feature map from the first

layer is then given by convolving each filter w_h^1 for $h = 1, ..., M_1$ with the input:

$$a^1(i,h) = (w_h^i \times x)(i) = \sum_{j=-\infty}^{\infty} w_h^1(j)x(i-j), \tag{3.5}$$

where $w_h^i \in \mathbb{R}^{1\times k\times 1}$ and $a^1 \in \mathbb{R}^{1\times N-k+1\times M_1}$. Note that since the number of input channels in this case is one, the weight matrix also has only one channel. Similar to the architecture of FF-ANN, this output is then passed through the non-linearity $h(\cdot)$ to give $f^1 = h(a^1)$. Furthermore, in each subsequent layer $l = 2, ..., L$ the input feature map, $f^{l-1} \in \mathbb{R}^{1\times N_{l-1}\times M_{l-1}}$, where $1 \times N_{l-1} \times M_{l-1}$ is the size of the output filter map from the previous convolution with $N_{l-1} = N_{l-2} - k + 1$, is convolved with a set of M_l filters $w_h^l \in \mathbb{R}^{1\times k\times M_l-1}, h = 1, ..., M_l$ to create a feature map $a_l \in \mathbb{R}^{1\times N_l\times M_l}$:

$$a^l(i,h) = (w_h^l \times f^{l-1})(i) = \sum_{j=\infty}^{\infty} \sum_{m=1}^{M_l-1} w_h^l(j,m) f^{l-1}(i-j,m). \tag{3.6}$$

The output of this is then passed through the non-linearity to give $f^l = h(a^l)$. The filter size parameter k thus controls the receptive field of each output node. Without zero padding, in every layer the convolution output has width $N_l = N_{l-1} - k + 1$ for $l = 1, ..., L$. Since all the elements in the feature map share the same weights this allows for features to be detected in a time-invariant manner, while at the same time it reduces the number of trainable parameters. The output of the network after L convolution layers will thus be the matrix f^L, the size of which depends on the filter size and number of filters used in the final layer. Depending on the training data, the weights in the network are trained to minimize the error between the output from the network f^L and the true output we are interested in.

3.2.2.2 Recurrent Neural Networks

In Recurrent Neural Networks (RNNs), the fundamental feature is that the network contains at least one feed-back connection, thus, the activations can flow around in a loop. In particular, this feature allows the network to have an infinite dynamic response to time series input data. RNNs architecture can have many different forms. A common type consists of a standard multilayer perception (MLP) plus added loops. These can exploit the powerful non-linear mapping capabilities of the MLP, coupled with an additional form of memory. Other architectures have more uniform structures, potentially with every neuron connected to all the others, and may also have stochastic activation functions. For simple architectures and deterministic activation functions, learning can be achieved using the back-propagation algorithm used in FF-ANN.

The simplest form of fully RNN is a MLP with the previous set of hidden unit activations feeding back into the network along with the inputs. A delay unit needs to be introduced to hold activations until they are processed at the next time step.

3.2.2.3 Infinite Impulse Response-Locally Recurrent Neural Network

The Infinite Impulse Response-Locally Recurrent Neural Network (IIR-LRNN) is a time-discrete network consisting of a global feed-forward structure of nodes interconnected by synapses which link the nodes of the i^{th} layer to those of the successive $(i+1)^{th}$ layer, $i = 0, 1, ..., M$ layer 0 being the input and M the output. Different from the classical FF-ANN, in an IIR-LRNN each synapse carries taps and feedback connections, meaning that the synapse inputs and outputs are fed back to the synapse with appropriate time delays. Specifically, each synapse of the IIR-LRNN contains an IIR linear filter whose transfer function can be expressed as ratio of two polynomials with poles and zeros representing the Auto Regressive (AR) and Moving Average(MA) part of the model, respectively. In particular, given a time series of data, the AR and MA part of the model is use for understanding and, predicting future values in the time series data. The task of the AR is to regress the variable on its own past values. On the other hand, the MA involves modelling the error term as a linear combination of error terms occurring simultaneously and at various times in the past. Similar to the classical FF-ANN, all data are normalized in a properly chosen range before being processed by the network. During the forward phase, at the generic time $t = 1, 2, ..., T$ the generic neuron $j = 1, 2, ..., N_k$ belonging to the generic layer $i = 1, 2, ..., M$ receives in input the quantity $y^i_{jl}(t)$ from neuron $l = 1, 2, ..., N^{i-1}$ of layer $i1$:

$$y^i_{jl}(t) = \sum_{p=0}^{L^{i-1}_{jl}} w^i_{jl(p)} \cdot x^{i-1}_l(t-p) + \sum_{p=1}^{I^i_{jl}} v^i_{jl(p)} \cdot y^i_{jl}(t-p). \tag{3.7}$$

The quantities $y^i_{jl}(t), l = 1, 2, ..., N^{i-1}$, are summed to obtain the net input $s^k_j(t)$ to the non-linear activation function $f^i(\cdot)$, of the j^{th} node, $j = 1, 2, ..., N^i$, of the i^{th} layer, $i = 1, 2, ..., M$:

$$s^i_j(t) = \sum_{l=0}^{N^{i-1}} y^k_{jl}(t). \tag{3.8}$$

Further, the output of the activation function gives the state of the j^{th} neuron of the i^{th} layer, $x^i_j(t)$:

$$x^i_j(t) = f^i(s^i_j(t)) \tag{3.9}$$

3.2.3 Uncertainty in Artificial Neural Network Computation

Subsequently, when using FF-ANNs or CNNs, and RNNs for regression problems, there are actually two types of prediction that one may want to obtain in correspondence of a given input. First, an estimate of the underlying non-linear function of interest. Second, an estimate of the target value itself. To these estimates it is crucial to associate their corresponding measures of confidence. However, there are several issues with the modelling of ANN for a problem which propagates additional uncertainties into the predicted quantity of interest from the network. These issues are further discussed in the following sections:

3.2.3.1 Uncertainty from Sampling Variability in Training Data Set

A portion of the total uncertainty in prediction values is attributable to the inherent uncertainty in the input data. From a probabilistic point of view, the data set used for training the network is only one of an infinite number of possible data sets which may be drawn within the given input volume and from the underlying statistical error distribution. In other words, this variability in the training data set is due to the variability in the sampling of the input vectors and in the random fluctuation of the corresponding target output.

3.2.3.2 Uncertainty from ANN Weight Parameters

Additionally, another source of uncertainty affecting the predicted value from the ANN arises as a result of the random initialization of the weight parameters of the ANN. In fact, when an ANN with a unique architecture is trained repeatedly with the same training data, different performing ANNs are being constructed. Consequently, this phenomenon gives rise to a model selection problem.

3.2.3.3 Uncertainty from the Model Structure

Furthermore, using an ANN for regression purpose involves finding an appropriate model structure $f(x)$ that describes the given example data. Generally, the relationship $f(x)$ is not known, however, it is often complex and non-linear, thus the network used to describe the relationship will have hundreds or even thousands of weight parameters. Ideally, achieving good performance from an ANN would involve selecting an ANN that has optimal complexity, where optimality is defined as the smallest network structure that adequately captures the underlying relationship. Contrarily, determining the optimal complexity is one of the most difficult tasks in designing an

ANN, as there exists no systematic method to ensure the optimal network will be chosen. The flexibility in ANN complexity primarily lies in selecting the appropriate number of hidden layer and neurons within these layers, which determine the number of weights in the model. Thus, a balance is required between having too few hidden nodes such that there are insufficient degrees of freedom to adequately capture the underlying relationship, and having too many hidden nodes such that the model fits to noise in the individual data points rather than the general trend underlying the data as a whole. The latter case is referred to as over-fitting, which is often difficult to detect but can significantly impair the performance of an ANN. In order to prevent over-fitting, cross validation (i.e. $k - fold$) during training is often used. However, apart from being more susceptible to over-fitting, a large ANN with many hidden nodes are inefficient to calibrate, the parameters and resulting predictions have a higher degree of associated uncertainty and it is more difficult to extract information about the modelled function from the parameters. Therefore, selection of the minimum number of necessary hidden nodes can be crucial to the performance of an ANN and its value as a prediction tool. Often, the commonly used method for selecting the number of hidden layer nodes is by trial and error approach, where a number of networks are trained, while the number of hidden nodes is systematically increased or decreased until the network with the best generalisability is found. Thereafter, the performance of the ANN is estimated by evaluating its out-of-sample performance, based on an independent test set using some goodness of fit measure, such as the root mean squared error (RMSE) or the coefficient of determination (R^2). However, this may not be practical if there are only limited available data, since the test data cannot be used for training. Furthermore, if the test data are not a representative subset, the evaluation may be biased.

3.3 Chapter Summary

ANN is a popular surrogate modelling technique which approximate the computational model by an inexpensive-to-evaluate function. The reason for selecting the ANN over other conventional surrogates is due to the fact that ANNs performs better when the input dimensional space is large. Although a classical multi-layer FF-ANN is good at approximating a complex non-linear function, it does not perform well for time series data. For this reason, dynamic ANNs that handle time series data adequately, such as CNN and RNN have evolved. Generally speaking, additional uncertainties are added to the prediction made by ANN as a result of variability in

the training data set, random initialization of the network weights, and the type of model structure chosen. The question then naturally arises, whether it is possible to quantify the uncertainties introduces in the ANN to ensure a robust reliable prediction. Hence, different approaches to quantify ANN uncertainties will be exploited in subsequent chapters.

Chapter 4

Robust Surrogate Models - Variability in Training Data

Computational models are commonly adopted in engineering practices due to their ability to replace costly and infeasible practical experiments. However, these models, suffer from high computational costs, thus, posing a challenge when performing simulation based reliability and sensitivity analyses. This is due to the large number of samples required for the robust estimation of the failure probability and its sensitivity indices. Additionally, in the reliability analysis of highly reliable systems such as those used in Nuclear engineering practices, the failure regions usually occupy a small region in the input domain, requiring a high number of model calls. Hence, surrogate models are built based on few training samples from the expensive models to reduce these computational cost. However, due to the variability in sampling training data from the input domain of the expensive models, important regions within the sample space can be missed. Thus, leading to an over/under estimation in the quantity of interest to be estimated by the surrogate model. Therefore, in this chapter, the bootstrap technique is adopted to deal with this type of problem. Furthermore, a novel stopping criterion is proposed for selecting the number of bootstrap models. To demonstrate the applicability and accuracy of this technique, it is adopted to compute the reliability and sensitivity indices of two analytical functions.

4.1 Background to the Bootstrap Technique

The bootstrap technique [59] is a distribution free inference method which requires no prior knowledge about the statistical distribution of the underlying population of interest. In particular, the basic idea is to generate a sample from the observed data by sampling with replacement from the original data set. Thereafter, an ensemble

of models is trained with the bootstrap samples. The quantity of interest focused at can then be estimated from this ensemble of bootstrap models. In this chapter, the quantity of interest focused at is the failure probability p_F and sensitivity indices S_i and T_i.

4.2 Succiant Theory of Reliability and Sensitivity Analyses

4.2.1 Reliability Analysis

The limit-state function of a system can simply be defined as a deterministic mapping from the z-dimensional input space to a one-dimensional output space:

$$G : \mathbf{x} \in D_{\mathbf{x}} \subset \mathbb{R}^z \rightarrow y = G(\mathbf{x}) \in \mathbb{R}, \tag{4.1}$$

where $\mathbf{x}$ is the z-dimensional state variables and $\mathbf{y}$ the performance variable (i.e. quantity of interest). $G(\mathbf{x})$ indicates if a realization $\mathbf{x} \in D_{\mathbf{x}}$ corresponds to the safe state ($G(\mathbf{x}) > 0$) or failed state ($G(\mathbf{x}) \leq 0$). In the context of probability theory, the failure probability, p_F, is defined as the probability that a realization $\mathbf{x} \in D_{\mathbf{x}}$ corresponds to a failed state in terms of the limit-state function $G(\mathbf{x})$:

$$p_F = \mathbb{P}(G(\mathbf{x}) \leq 0) = \int_{D_f} f_X(\mathbf{x}) d\mathbf{x}, \tag{4.2}$$

where $D_f = \mathbf{x} \in D_{\mathbf{x}} : G(\mathbf{x}) \leq 0$ is the failure region and $f_X(\mathbf{x})$ is the joint probability density function of the state variables X. As Eq.(4.2) is analytically intractable due its multidimensional nature, MC simulation (see [60]) allows one to numerically compute the estimate of the failure probability p_F, considering a large sample of size N :

$$\hat{p}_F = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}_{G(\mathbf{x}) \leq 0}(x_i), \tag{4.3}$$

where $\mathbb{I}_{G(\mathbf{x}) \leq 0}$ is the indicator function for failure such that $\mathbb{I} = 1$ for $G(\mathbf{x}) \leq 0$ and $\mathbb{I} = 0$ otherwise.

4.2.2 Sensitivity Analysis

Global sensitivity analysis methods are based on the analysis of variance (ANOVA), which estimates the fractional contribution of each uncertain input parameter to the variance of the quantity of interest. The first order and total effect indices are mostly

used in this regard, where each index is computed by evaluating a multidimensional integral via MC simulation. Particularly, this approach decomposes the variance of the output of interest into fractions that can be attributed to each of the input parameters of the model. The total variance of the output quantity of interest is expressed as:

$$Var(y) = \sum_{i=1}^{d} V_i + \sum_{i<j}^{d} V_{ij} + \cdots + V_{1,2,\cdots,d}, \tag{4.4}$$

where d denotes the number of model dimension, V_i the partial variance for single parameters and, V_{ij} represents interactions between parameters. Mathematically, V_i and V_{ij} are defined as:

$$V_i = Var_{x_i}(E_{x\sim i}(y|x_i)), \tag{4.5}$$

and

$$V_{ij} = Var_{x_{ij}}(E_{x\sim ij}(y|x_{ij})). \tag{4.6}$$

The first order sensitivity measure is defined as:

$$S_i = \frac{V_i}{Var(y)}. \tag{4.7}$$

Equation 4.7 measures the effect of varying the input parameters of the model, without considering interactions between parameters of the model. Similarly, the second order sensitivity measure that measures the sensitivity due to interactions between parameters of the model is defined as:

$$S_{ij} = \frac{V_{ij}}{Var(y)}. \tag{4.8}$$

Finally, the total effect sensitivity measure, T_i, that measures the contribution of a single parameter as well as interactions between parameters in the model is mathematically defined as:

$$T_i = 1 - \frac{Var_{x\sim i}[E_{x\sim i}(y|x_{\sim i})]}{Var(y)}. \tag{4.9}$$

4.2.3 Modelling of Artificial Neural Network for Reliability and Sensitivity Analysis

As previously stated, a setback on the use of MC simulation methods to estimate the value of p_F, S_i and T_i, is the large number of model calls required for accuracy. Therefore, ANNs are being used in place of expensive models to quicken the required analyses. The construction ANN models requires a set of real-valued input/output data pairs $D_{train}(\mathbf{x}, \mathbf{y})$ of size N_{train} generated according to a signal plus noise model

$\mathbf{y} = \mu(\mathbf{x}) + \epsilon$. Where $\mathbf{y}$ is the observed performance generated from the expensive model, $\mathbf{x}$ is the independent state variables sampled from a joint probability density $\Omega(\mathbf{x})$, ϵ is independent, identically distributed (iid) noise sampled from a density $\Psi(\epsilon)$ (not necessarily Gaussian) having mean of 0 and variance σ^2, and $\mu(\mathbf{x})$ the unknown function that is needed to be approximated, by finding an approximation $\hat{\mu}(\mathbf{x})$ from $D_{train}(\mathbf{x}, \mathbf{y})$. A priori assumption can be made about the functional form of $\mu(\mathbf{x})$. However, since a parametric function class is usually unknown, a non-parametric regression approach must be resorted to. Using the non-parametric approach, one constructs an estimate, $\hat{\mu}(\mathbf{x})$, of $\mu(\mathbf{x})$ from a large class of functions, Υ, known to have good approximation properties. The class of approximation functions usually contains a set of estimators $f(\mathbf{x}, w^\alpha) \subset \Upsilon$ for which the elements of each subclass $f(\mathbf{x}, w^\alpha)$ are continuously parametrized by a set of p weights $w^\alpha; \alpha = 1, 2, ..., p$. The gradient decent algorithm [61] which is used to minimize the cost function $J(w^\alpha)$ of $f(\mathbf{x}, w^\alpha)$ is defined as:

$$J(w^\alpha) = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} (y_i - \hat{y}_i)^2, \tag{4.10}$$

by finding a set of weights w^α such that for any given input, the cost function is sufficiently small.

4.2.4 Variability in Training Data Set

From a probabilistic point of view, the training data set used for training the ANN $f(\mathbf{x}, w^\alpha)$ is only one of the possible infinite sets that can be drawn from the underlying population. As stated in Chapter 2, the variability in the training data set D_{train} is as a result of the variability in the sampling algorithm used to sample from the input space and the random fluctuation of the corresponding target output. Subsequently, each possible training set can give rise to a different set of network weights w^α. Thus, if only the training data set D_{train} is used to train $f(\mathbf{x}, w^\alpha)$, the variance term of the quantity of interest will be relatively high. Hence, in this chapter, the bootstrap technique is used to quantify the variance and further reduce the bias term in the quantity estimated from the surrogate model.

4.2.5 Adaptive Bootstrap Algorithm for Surrogate Models

In this section, an adaptive bootstrap algorithm is developed to quantify the effect of variability in training data set. In particular, the algorithm uses a stopping condition that determines the number of bootstrap models to be constructed, rather than

selecting a predetermined number of bootstrap models to be constructed. The procedures for the adaptive bootstrap algorithm used to assess the variance propagated by the variability in training data set are given in the following steps.

Step 1: Generate a set D_{train} of input-output training data by sampling N_{train} independent input parameters values x_p, $p = 1, 2, ..., N_{train}$, and calculating the corresponding set of N_{train} output vectors $y_p = \mu_y(x_p)$ from the expensive model. Experimental design algorithms such as Latin Hypercube Sampling (LHS) or other sophisticated experimental design algorithms can be adopted to select the input vectors $x_p, p = 1, 2, ..., N_{train}$.

Step 2: Construct the surrogate model $f(\mathbf{x}, w^\alpha)$ (i.e. ANN) on the basis of the entire training data set D_{train} in order to obtain an easy to evaluate surrogate of the expensive model represented by the unknown non-linear deterministic function, $\mu_y(\mathbf{x})$.

Step 3: Use the surrogate model, $f(\mathbf{x}, w^\alpha)$, in place of the high fidelity model, to provide a point estimate $\hat{Q}$ of the quantity Q, i.e., $\hat{p}_F$, S_i, and T_i. Specifically, draw N_T new samples from input vectors $x_r, r = 1, 2, ..., N_T$, from the corresponding probability distributions and feed the surrogate model $f(\mathbf{x}, w^\alpha)$ with them. Thereafter, use the corresponding output vectors $y_r = f(x_r, w^\alpha), r = 1, 2, ..., N_T$, to calculate the estimate $\hat{Q}$ for Q. Since the surrogate model $f(\mathbf{x}, w^\alpha)$ can be evaluated quickly, this step is computationally cheap even if the number, N_T, of model estimations is very high (e.g., $N_T = 10^5 - 10^6$).

Step 4: Construct an ensemble of B surrogate models $\{f_b(\mathbf{x}, w_b^\alpha), b = 1, 2, ...B\}$ based on a criterion that selects the number of B models to be constructed in order to calculate an estimate $\hat{Q}_b, b = 1, 2, ..., B$, for the quantity Q of interest. By so doing, a bootstrap-based empirical probability distribution for the quantity Q is produced which is the basis for the construction of the corresponding confidence intervals. In particular, repeat the following steps for $b = 1, 2, ..., B$:

a Generate a bootstrap data set $D_{train,b}\{(x_{p,b}, y_{p,b}), p = 1, 2, ..., N_{train}, b = 1, 2, ..., B\}$, by performing random sampling with replacement from the original data set $D_{train} = (x_p, y_p), p = 1, 2, ..., N_{train}$ of N_{train} input-output patterns. The data set $D_{train,b}$ is thus constituted by the same number N_{train} of input-output patterns drawn among those in D_{train} although, due to the sampling with replacement, some of the patterns in D_{train} will appear more than once in $D_{train,b}$, whereas some will not appear at all.

b Build ensemble of bootstrap surrogate models $f_b(\mathbf{x}, w_b^\alpha), b = 1, 2, ..., B$, on the basis of the bootstrap data $D_{train,b} = \{(x_{p,b}, y_{p,b}), p = 1, 2, ..., N_{train}\}$.

c Use the bootstrap surrogate models $f_b(\mathbf{x}, w_b^\alpha)$, in place of the expensive model, to compute a point estimate $\hat{Q}_b$ of the quantity of interest Q. Note that for a correct quantification of the confidence intervals, the estimate $\hat{Q}_b$ must be based on the same input and output vectors x_r and $y_r, r = 1, 2, ..., N_T$, respectively.

Step 5: Calculate the Bootstrap Bias Corrected (BBC) point estimate $\hat{Q}_{BBC}$ for Q as $\hat{Q}_{BBC} = 2\hat{Q} - \hat{Q}_{boot}$ where $\hat{Q}$ is the estimate obtained with the surrogate model $f(\mathbf{x}, w^*)$ trained with the original data set D_{train} and $\hat{Q}_{boot}$ is the average of the B estimates $\hat{Q}_b$ obtained with the B surrogate models $f_b(\mathbf{x}, w_b^\alpha), b = 1, 2, ..., B$.

$$Q_{boot} = \frac{1}{B} \sum_{b=1}^{B} \hat{Q}_b \tag{4.11}$$

The BBC estimate $\hat{Q}_{BBC}$ is taken as the point estimate for Q. The expression of $\hat{Q}_{BBC}$ can be proven from the theory that if there is a bias in the bootstrap average estimate $\hat{Q}_{boot}$ compared to the estimate $\hat{Q}$ obtained with the single regression model $f(\mathbf{x}, w^\alpha)$, then the same bias exists in the single estimate $\hat{Q}$ compared to the true value Q of the quantity of interest. Thus, in order to obtain an appropriate, i.e. bias-corrected, estimate $\hat{Q}_{BBC}$ for the quantity of interest Q, the estimate $\hat{Q}$ must be adjusted by subtracting the corresponding bias $(\hat{Q}_{boot} - \hat{Q})$: as a consequence, the final, bias corrected estimate $\hat{Q}_{BBC}$ is $\hat{Q}_{BBC} = \hat{Q} - (\hat{Q}_{boot} - \hat{Q}) = 2\hat{Q} - \hat{Q}_{boot}$

Step 6: Calculate the two-sided Bootstrap Bias Corrected (BBC) confidence interval for the BBC point estimate doing the following:

a Order the bootstrap estimates $\hat{Q}_b, b = 1, 2, ..., B$ by increasing values, such that $\hat{Q}_{(i)} = \hat{Q}_b$ for some $b = 1, 2, ..., B$, and $\hat{Q}_{(1)} < \hat{Q}_{(2)} < ... < \hat{Q}_{(b)} < ... < \hat{Q}_{(B)}$.

b Identify the $100 \cdot \alpha/2^{th}$ and $100 \cdot (1 - \alpha/2)^{th}$ quantiles of the bootstrapped empirical probability distribution of Q as the $[B \cdot \alpha/2]^{th}$ and $[B \cdot (1-\alpha/2)]^{th}$ elements $\hat{Q}_{([B \cdot \alpha/2])}$ and $\hat{Q}_{([B \cdot (1-\alpha/2)])}$, respectively ascending order $\hat{Q}_{(1)} < \hat{Q}_{(2)} < \cdots < \hat{Q}_{(b)} < \cdots < \hat{Q}_{(B)}$

c Calculate the confidence intervals of $\hat{Q}_{BBC}$ as:

$$\underline{\hat{Q}}_{BBC} = \hat{Q}_{BBC} - (\hat{Q}_{boot} - \hat{Q}_{([B\cdot\alpha/2])}) \tag{4.12}$$

$$\overline{\hat{Q}}_{BBC} = \hat{Q}_{BBC} + (\hat{Q}_{([B\cdot(1-\alpha/2)])} - \hat{Q}_{boot}) \tag{4.13}$$

Step 7: Check the stopping criterion of the algorithm based on Equation 4.14. If condition is met, go to next step, else, return to Step 4 and build $B+1$ ANNs.

Step 8: Stop the algorithm.

4.2.5.1 Criterion for Selecting the Number of Bootstrap Models to be Constructed

Here, a criterion used to select the number of bootstrap models B to be constructed is proposed based on the magnitude of BBC estimate from the bootstrap estimate. The criterion is given as follows:

$$argmin\{\mathbf{E}[\hat{Q}_{BBC}] \approx y_{train} : \underline{\hat{Q}}_{BBC} \leq y_{train} \leq \overline{\hat{Q}}_{BBC}\} \tag{4.14}$$

The flow chart of the above algorithm is given in Figure 4.1.

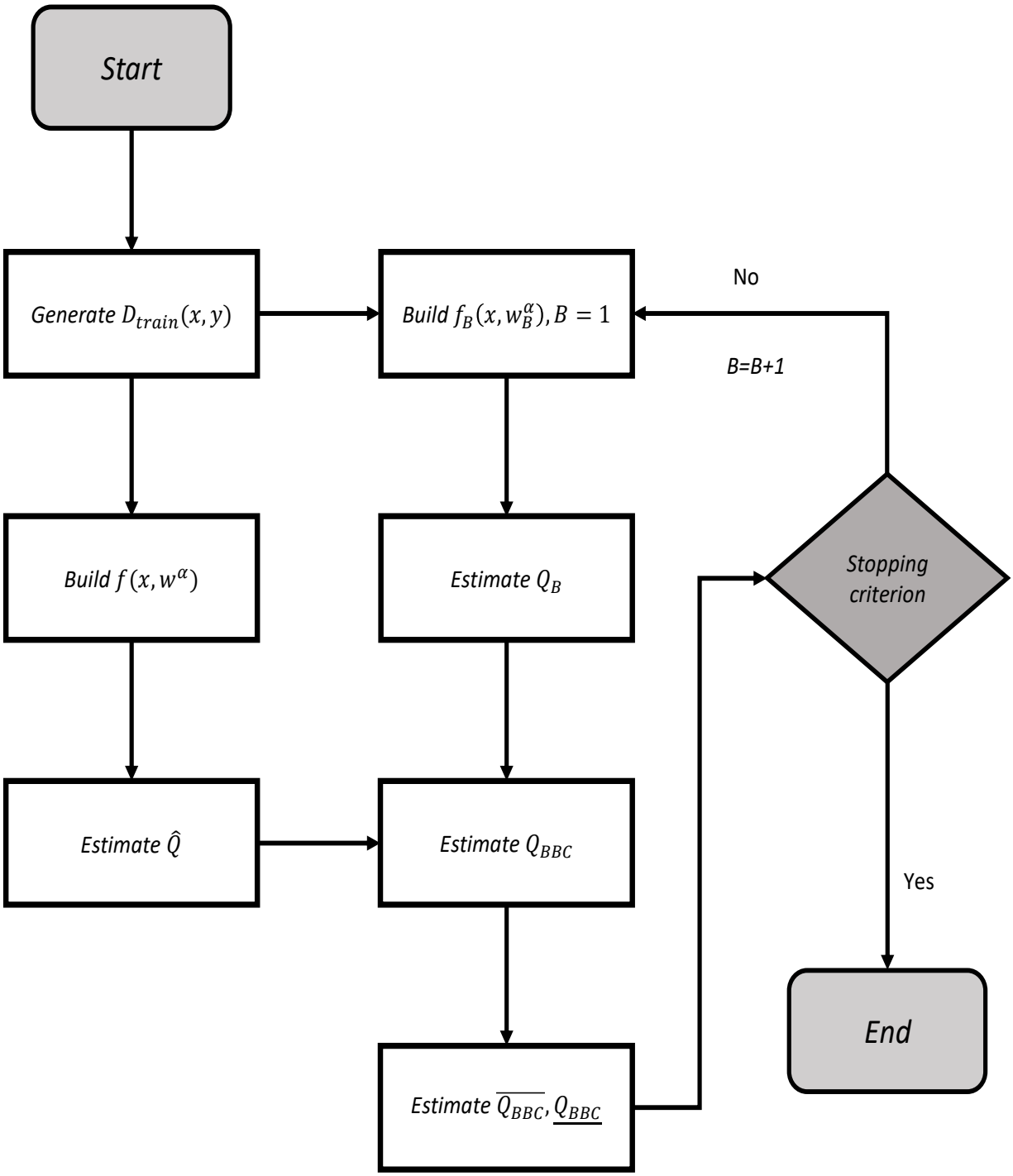


Figure 4.1: Flow Chart for Adaptive Bootstrap Algorithm

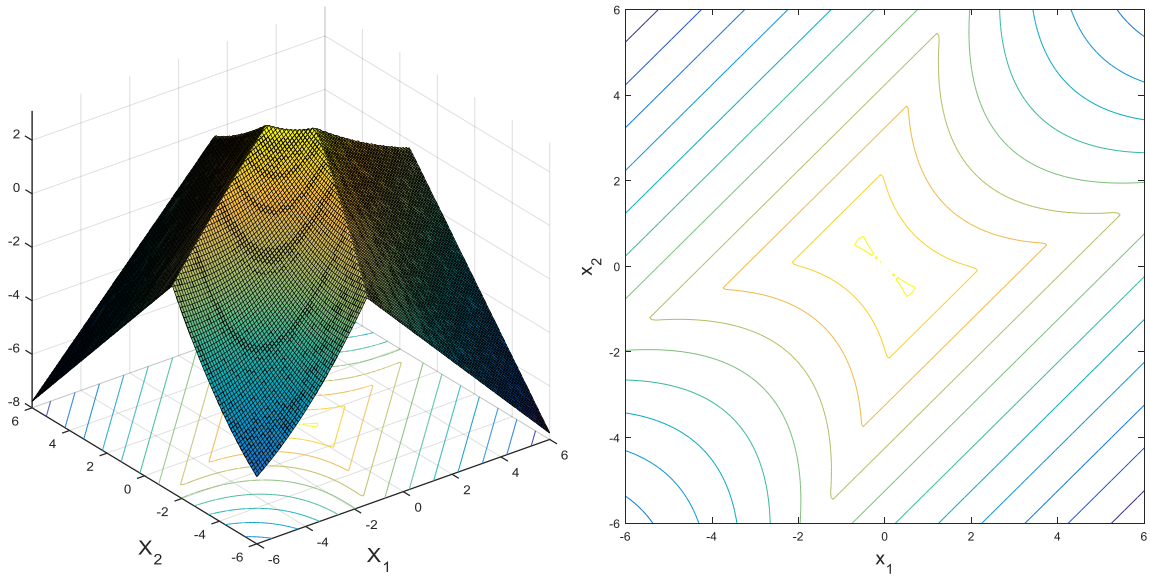
4.3 Case Study

4.3.1 Case Study 1: The Four Branch Function

The four-branch function is a common benchmark in structural reliability analysis that describes the failure of a series system with four distinct component limit states. Its mathematical formulation is given as:

$$f_1(\mathbf{x}) = min \begin{Bmatrix} 3 + 0.1(x_1 - x_2)^2 - \dfrac{x_1 + x_2}{\sqrt{2}} \\ 3 + 0.1(x_1 - x_2)^2 + \dfrac{x_1 + x_2}{\sqrt{2}} \\ (x_1 - x_2) + \dfrac{6}{\sqrt{2}} \\ (x_2 - x_1) + \dfrac{6}{\sqrt{2}} \end{Bmatrix}$$

where the input variables are modelled by two independent Gaussian random variables $x_i = \mathcal{N}(0, 1)$. The failure event is defined as $f_1(\mathbf{x}) \leq 0$, i.e. the failure probability is $p_F = \mathbb{P}(f_1(\mathbf{x}) \leq 0)$. The surface and contour plot of the four-branch function is given in Figure 4.2.



(a) Surface Plot (b) Contour Plot

Figure 4.2: Limit State - Four branched Function

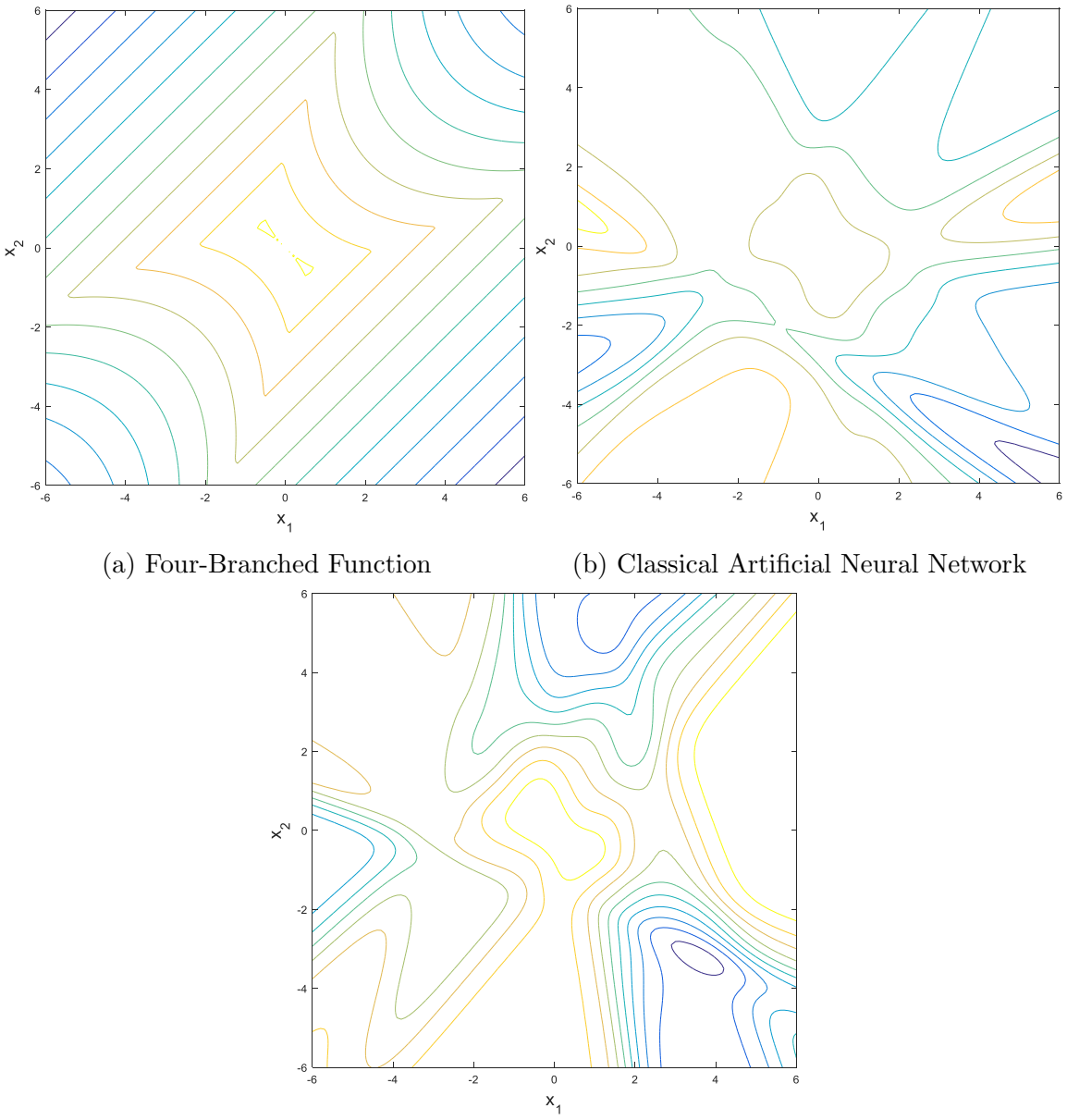
4.3.1.1 Analysis

To train the bootstrap ANNs, $N_{train} = 200$ training samples have been generated via LHS design technique. Then, the algorithm described in Section 4.2.5 have been

followed converging after $B = 300$ iterations.

4.3.1.2 Results

Here, the behaviour of the different types of limit states are shown in Figure 4.3. Figure 4.3(a) shows the limit state estimated from the real model, while, Figures 4.3(b) and 4.3(c) show the limit-state function estimated from classical ANN and bootstrap ANNs respectively.



(a) Four-Branched Function

(b) Classical Artificial Neural Network

(c) Bootstrap Artificial Neural Network

Figure 4.3: Four-Branched Function - Visual Composition of Bootstrap Artificial Neural Network

As shown in Figure 4.3(b), the classical ANN underestimates the failure domain due to the fact that the red contour is not visible. On the other hand, the bootstrap ANN (Figure 4.3(c) gives a reasonable estimate of the failure domain from $B = 300$ iterations (i.e. red contour is visible). Furthermore, the failure probability $\hat{p}_F$ has been estimated adopting the bootstrap ANNs, using $N = 10^5$ MC samples for the estimation of $\hat{p}_F$. The estimate of $\underline{\hat{Q}}_{BBC}$, $\hat{Q}_{BBC}$, and $\overline{\hat{Q}}_{BBC}$ are found to be $0.2723 \times 10^{-3}, 0.2761 \times 10^{-3}, 0.2823 \times 10^{-3}$ respectively. The reference value of $\hat{p}_F = 0.2759 \times 10^{-3}$ has been estimated using $N = 10^6$ MC samples for a robust comparison. From the results obtained, the bootstrapped ANNs turn out to be quite reliable and robust, providing BBC point estimates very close to the reference value of $\hat{p}_F$. Table 4.1 shows the percentage of true value of the validation data that falls inside the confidence bounds computed using the bootstrap approach. Also in the same table, error bounds have been estimated from 300 trials with classical ANN. From the results, 99.5% of the validation samples falls within the BBC confidence intervals at $B = 300$. However, using 300 classic ANN to compute error bounds $\min[\hat{Q}]$ and $\max[\hat{Q}]$, only 55.5% of the validation samples fall within the bounds.

Table 4.1: Accuracy of Method for Four-Branched Function

Approach	Percentage Accuracy	Computational Time
Classic ANN (300 trials)	55.5%	0.0259s
B=50	75%	0.1075s
B=100	80%	0.215s
B=200	86%	0.335s
B=300	99.5%	0.432s
B=500	99.5%	0.767s

Bootstrap Prediction Intervals Estimation

To assess the quality of the estimated bootstrapped confidence intervals $\underline{\hat{Q}}_{BBC}, \overline{\hat{Q}}_{BBC}$ the coverage probability and the interval width is considered. The coverage probability is evaluated by the Prediction Interval Coverage Probability ($PICP$) metric, which quantifies the probability that the estimated intervals contain the true value. This metric read as follow [62]:

$$PICP = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} C_i \text{ with } \begin{cases} C_i = 1, & \text{if } Q_i \in [\underline{\hat{Q}}_{BBC}, \overline{\hat{Q}}_{BBC}] \\ C_i = 0, & \text{otherwise} \end{cases} \tag{4.15}$$

The interval width is evaluated by the Normalized Mean PI Width ($NMPIW$) metric which quantifies the average intervals width normalized with respect to the target

value. It reads as follow [62]:

$$NMPIW = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{\underline{\hat{Q}}_{BBC} - \overline{\hat{Q}}_{BBC}}{\hat{Q}_{BBC}} \tag{4.16}$$

The objectives in these analyses are to have the tightest intervals with the largest probability of containing the real value Q. Thus, the aim is to maximise the $PICP$ and minimise the $NMPIW$ simultaneously.

Table 4.2: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (300 trials)	0.92	0.45
$B = 50$	0.90	0.40
$B = 100$	0.92	0.41
$B = 200$	0.92	0.42
$B = 300$	0.94	0.39
$B = 500$	0.94	0.39

Table 4.2 shows the $PICP$ and $NMPIW$ values obtained from different bootstrapped ANNs. Table 4.2 also shows the $PICP$ and $NMPIW$ values obtained from a combination of 300 classical ANNs. As shown in Table 4.2, it can be seen that the best $PICP$ and $NMPIW$ values are obtained at $B = 300$. However, the $PICP$ and $NMPIW$ values estimated from the interval obtained from 300 classical ANNs are relatively lower. Thus, the bootstrap approach is robust enough to produce better intervals that quantify the uncertainties and capture the true prediction of the quantity of interest. Though, using the bootstrap approach is expensive as it involves training a large number of models, parallelization strategies can be employed to quicken the analyses.

4.3.2 Case Study 2: The Ishigami Function

Here, Ishigami function is used as a benchmark model to test the robustness of the bootstrap technique for predicting sensitivity indices. The Ishigami function is represented by the following equation:

$$f_2(\mathbf{x}) = sin(x_1) + asin^2 x_2 + bx_3^4 sin(x_1) \tag{4.17}$$

In this example, the numerical values chosen for a and b are 2 and 1 respectively. The parameters $x_i, i = 1, 2, 3$ are uniformly distributed in the interval of $-\pi$ and π.

The objective of this example is to train an ANN to replace the model for sensitivity analysis, then quantify the model uncertainties in the sensitivity indices estimates due to sampling variability in the ANN training data.

4.3.2.1 Analysis

Training samples $D_{train}(x, y)$ of size $N_{train} = 200$ have been generated via Latin hypercube algorithm [22] from the Ishigami function. Thereafter, $B = 500$ ANNs have been constructed based on the adaptive bootstrap algorithm. Furthermore, the sensitivity indices of each parameter have been computed adopting 10^4 samples for each of the good performing bootstrap ANN, and combined based on the listed steps in Section 4.2.5. The result of this analysis is shown in Table 4.3 and 4.4.

4.3.2.2 Results

The following displays the results applying the bootstrap technique to the Ishigami function.

Table 4.3: First Order Sobol' Indices

$Parameter$	$\underline{\hat{Q}}_{BBC}$	$\hat{Q}_{BBC}$	$\overline{\hat{Q}}_{BBC}$	$Reference$
x_1	0.3528	0.4072	0.4591	0.3919
x_2	0.0001	0.0002	0.0005	0.0002
x_3	0.0019	0.0097	0.0357	0.0112

Table 4.4: Total Effect Indices

$Parameter$	$\underline{\hat{Q}}_{BBC}$	$\hat{Q}_{BBC}$	$\overline{\hat{Q}}_{BBC}$	$Reference$
x_1	0.9609	0.9805	0.9892	0.9984
x_2	0.004	0.0005	0.0118	0.0051
x_3	0.5357	0.5870	0.6419	0.6079

Note that the reference sensitivity indices in Table 4.3 and 4.4, have been obtained with the same number of model evaluations from the Ishigami function, to provide a robust reference for the comparisons. From the results in Tables 4.3 and 4.4, there is a good match between the results from the Ishigami function and those from the bootstrapped ANNs, this leads us to assert that the accuracy in the estimates from bootstrap ANNs can be considered satisfactory for the need of percentile estimation of sensitivity indices. Also, it can be seen that the BBC estimates from the ANNs

are much closer to the reference results. Moreover, the uncertainty associated with the bootstrapped ANNs are significantly lower and captures the reference estimate.

Table 4.5: Accuracy of Method for Ishigami Function

Approach	Accuracy	Computational Time
Classic ANN (500 trials)	54.5%	0.026s
B=50	73%	0.1082s
B=100	82%	0.230s
B=200	88%	0.355s
B=300	98%	0.475s
B=500	98%	0.785s

Table 4.5 shows the percentage of true value of the validation data that falls inside the confidence bounds computed using the bootstrap approach. Also in the same table, error bounds have been estimated from 500 trials with classical ANN. From the results, 98% of the validation samples falls within the BBC confidence intervals at $B = 300$. However, for 500 trials with classical ANN, only 54.5% of the validation samples fall within the bounds. Furthermore, the $PICP$ and $NMPIW$ values of these ANNs are shown in Table 4.6.

Table 4.6: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (500 trials)	0.92	0.45
B=50	0.93	0.42
B=100	0.93	0.40
B=200	0.94	0.41
B=300	0.95	0.40
B=500	0.95	0.40

4.4 Chapter Summary

In this chapter, Artificial Neural Networks (ANNs) have been used in the task of estimating, in a fast and efficient way, the failure probability and sensitivity indices of non-linear analytical functions. However, using ANNs for this purpose introduces additional variability and bias to the quantity of interest being focused on, due to sampling variability in the training data set extracted from the high fidelity model. Therefore, a bootstrap technique has been adopted to deal with this problem. The bootstrap method is used to estimate confidence intervals on the quantities computed

(i.e. failure probability and sensitivity indices). This uncertainty quantification is of paramount importance in safety critical applications, in particular when few training data are used. In this regard, bootstrapped ANNs have been shown to produce results that are closer to the reference value in the analytical examples tested. On this basis, bootstrapped ANNs can be considered effective in the estimation of the failure probability and sensitivity indices (while quantifying the uncertainty associated with the results) because they provide more accurate (i.e., estimates are closer to the true values) and precise (i.e., confidence intervals are narrower) estimates. On the other hand, the computational time required by bootstrapped ANNs is in the high order of magnitude, due to the training algorithm for building the structure of complex ANN and the number of bootstrap models to be trained, however, parallelization strategies can be employed to reduce this computational cost.

Chapter 5

Robust Surrogate Models - Random Model Parameters

It is known that the back-propagation algorithm in an ANN can be viewed as the optimization of the error function with respect to the weights. A local optimization technique is usually employed for training an ANN and as a consequence the training algorithm usually get trapped at a local minimum. Furthermore, the particular local minimum will determine the quality of the ANN solution. On the one hand, if the minimum is close to the global one the performance will be acceptable and the training successful. Additionally, there are minima that result in poorly trained networks and unsuccessful convergence. The factors that determine the final local minimum are mainly the particular weight initialization and the training algorithm. Furthermore, the weight initialization influences the speed of convergence, the probability of convergence and the generalization. Notably, when an ANN with a selected architecture is trained multiple times from the same training data, different performing ANNs are produced, due to random initialization of the weight parameters by the back-propagation algorithm in each ANN. Hence, a model selection problem is introduced. Consequently, cross-validation methods (i.e. $k-fold$) are frequently used to select the best performing ANN by computing a performance metric such as the coefficient of determination R^2 and selecting the network with the best metric. However, there are two disadvantages to such an approach. First, all the effort used in training a set of identical network is wasted as only one network is selected and the rest discarded. Second, due to the random noise component within the data used to validate the networks, the network which had best performance on the validation set might not be the one with the best performance on a new test data set. Subsequently, these drawbacks can be overcome by combining the ensemble of ANNs together to form a committee. The importance of such an approach is that it can lead to significant improvements in the

predictions on new data, while involving little additional computational effort. Thus, in this chapter, we propose an approach to improve the prediction made by ANN. Note that the approach proposed in this chapter have been published in [48]. The approach is based on a systematic combination of identical trained ANNs, by coupling the Bayesian framework and model averaging. Additionally, the uncertainties of the robust prediction derived from the approach are quantified in terms of confidence intervals. To demonstrate the applicability of the proposed approach, two synthetic reliability examples are tested with the approach.

5.1 Background Theory of Proposed Approach

The proposed approach in this chapter is aimed towards improving the predctive performance of an ANN used for a variety of applications. Thus, the underlining idea behind the proposed approach is to construct a set of ANNs (i.e. same architecture) based on the same training data $D_{train}(x, y)$. By so doing, a distribution of identical ANNs whose error functions are trapped in different local minima is created. The major highlight of this approach is that the solution space of the error function is exploited as many times as possible with the possibility of locating a global minima on the error surface. Furthermore, Bayesian statistics is adopted to evaluate the posterior probability of each of the trained ANN based on their likelihood to predict the training data. Thereafter, a model averaging technique (adjustment factor approach see [63]) is used to combine the prediction made by the ensemble of ANNs to yield a robust prediction that outperforms the prediction made by a single ANN. Finally, the model uncertainty as a result of the random model parameters are propagated to the predicted quantity and quantified in terms of confidence intervals.

5.1.1 Bayesian Model Selection for Identical Trained Artificial Neural Networks

Given a set $\mathcal{S}$ of M identical competing ANNs $\mathcal{S} = \{N_1, N_2, ..., N_M\}$ trained with same data $D_{train}(x, y)$, Bayesian statistics can be used to infer the posterior probability of the m^{th} ANN in $\mathcal{S}$, i.e. the N_m in the set, such that:

$$P(N_m|D_{train}) = \frac{P(D_{train}(x,y)|N_m)P(N_k)}{\sum_{q=1}^{M} P(D_{train}(x,y)|N_q)P(N_q)} \tag{5.1}$$

where $P(D_{train}(x, y)|N_k)$ is the likelihood of training data $D_{train}(x, y)$ for the N_m ANN, and $P(N_m)$ is the prior probability of N_m, which is the ANN probability evaluated before observing training data $D_{train}(x, y)$. The prior ANN probability $P(N_m)$

can be specified depending on the existing prior knowledge about the credibility of ANN N_m, or it can be given as a uniform probability, $P(N_m) = 1/M$, if no additional information is provided. The advantage of assigning uniform prior probability to $P(N_m)$ is that the difficulty of estimating the prior probability numerically is avoided. The likelihood $P(D_{train}(x, y)|N_m)$ may be thought of as the probability of observing the training data $D_{train}(x, y)$ under ANN N_m. It supplies a relative measure of how well the ANN N_m is supported by the training data $D_{train}(x, y)$. Since the denominator in Eq.(5.1) is common for all the ANNs, the posterior ANN probability is proportional to prior probability and the likelihood. The likelihood of each ANN is evaluated by measuring the degree of agreement between the training data $D_{train}(y)$ and the response $\hat{y}$ for each ANN. Hence, a probabilistic relationship between training data $D_{train}(x, y)$ and ANN predictions $\hat{y}$ involving uncertainty can be described. Typically, the bias function and noise are included as parts of the probabilistic relationship to match ANN predictions with training data. The bias function captures the discrepancies between the expensive model responses and predictions made by the ANN. The noise is usually assumed to be independent and identically distributed normal random variable with a mean of zero [64]. Various authors [65–67] have used the Bayesian statistics to quantify the uncertainty in the bias function modelled as a Gaussian process. In their respective work, a mathematical formulation that combines bias function associated with the ANN and noise from training data is utilized to describe the probabilistic relationship between the training data $D_{train}(x, y)$ and ANN predictions $\hat{y}$. The mathematical formulation of this probabilistic relationship is given by the following equation:

$$D_{train}(y) = \hat{y} - \varepsilon, \tag{5.2}$$

where ε is a random variable that covers both bias associated with the ANN prediction $\hat{y}$ and the noise in the response training data $D_{train}(y)$. ε is assumed to be an independent identically distributed random variable with a mean μ of zero. The use of ε with zero mean does not shift ANN prediction $\hat{y}$. This reflects the fact that $\hat{y}$ is the most probable prediction value for the ANN. The bias function is not included as a separate term in the probabilistic relationship. This is due to the fact that introducing a separate bias function results in shifting the prediction $\hat{y}$ of the ANN from the initially predicted value. The likelihood $P(D_{train}(x, y)|N_m)$ of training data $D_{train}(x, y)$ for ANN N_m is evaluated by observing where the training data points $D_{train}(y)$ are located in the distribution of $\hat{y}$ estimated by N_m. The procedures to estimate the distribution $P(\hat{y}|N_m)$ of N_m and the likelihood $P(D_{train}(x, y)|N_m)$ is

given. First, the uncertainty in errors of predictions $\hat{y}$ made by N_m is quantified by introducing an assumption that the prediction errors are independent and identically distributed normal random variable with a mean μ of zero. The error of the prediction of the m^{th} network is represented by the following:

$$\varepsilon_{mi} = D_{train}(y_i) - \hat{y}_i, \varepsilon_{mi} \sim N(0, \sigma_m^2), i = 1, 2, ..., N_{train}, \tag{5.3}$$

where $D_{train}(y_i)$ is the i^{th} training response output data, $\hat{y}_i$ the prediction of the training data made by N_m, σ_m^2 the variance of prediction error ε_{mi}, and N_{train} the number of samples in the training data. The prediction error ε_{mi} measured is considered to be a random sample from a normal distribution with a mean (μ) of zero and variance σ_m^2. Using the principle of maximum likelihood estimation (MLE) (see [68]), the variance σ_m^2 for N_m can be estimated as:

$$\sigma_m^2 = \frac{1}{N} \sum_{i=1}^{N} \varepsilon_{mi}^2. \tag{5.4}$$

Secondly, the predictive distribution $P(\hat{y}|N_m)$ of response $\hat{y}$ under model N_m is created by including the prediction error obtained in the previous step into the prediction of $\hat{y}$ made by N_m. This predictive distribution is defined by the following equation:

$$P(\hat{y}|N_m) = D_{train}(y) + \varepsilon_{mi}. \tag{5.5}$$

Lastly, assuming that the residuals between the training data $D_{train}(x, y)$ and N_m output $\hat{y}$ are normally and independently distributed with a mean of zero and constant variance σ_m^2, the likelihood function $P(D_{train}(x, y)|N_m)$ is approximated by:

$$P(D_{train}(x, y)|N_m) \approx \frac{1}{\sqrt{2\pi\sigma_m^2}} \frac{1}{N} \sum_{i=1}^{N} \exp\{\frac{-[y_i - \hat{y}_{mi}]^2}{2\sigma_m^2}\}. \tag{5.6}$$

5.1.2 Robust Artificial Neural Network Prediction

Subsequently, to obtain a robust prediction from an ANN, the estimates made by all the subsequent trained ANNs are combined using a model averaging technique. Specifically, the adjustment factor approach (see [63]) is adopted and combined with Bayesian statistics. With this approach, the ANN having the highest posterior probability is used in conjunction with other respective ANNs trained to correct the bias estimate predicted by a single ANN. The adjustment factor is evaluated by assuming the error between the prediction of all the subsequent trained ANNs and the training

data are normally distributed. For the quantification of the robust value, the posterior probability computed for each ANN is used as a weighting. A distribution from the response predicted by the ANNs is created by introducing the adjustment factor A_f which is characterized by a normal distribution. The robust ANN prediction can be obtained from the following equation:

$$y_{robust} = \hat{y}^* + A_f, \tag{5.7}$$

where $\hat{y}^*$ represents the point estimate of the best ANN in the set with the highest probability, A_f represents the adjustment factor, and y_{robust} represent the robust prediction which also incorporates the model uncertainty. Since the adjustment factor A_f is assumed to be a normal distribution, the expected value and variance of the adjustment factor A_f is given by the following relationships:

$$\mathbf{E}[A_f] = \sum_{m=1}^{M} P(N_m|D_{train})(\hat{y}_m - \hat{y}^*), \tag{5.8}$$

$$\mathbf{Var}[A_f] = \sum_{k=1}^{M} P(N_m|D_{train})(\hat{y}_m - \mathbf{E}[y_{robust}])^2. \tag{5.9}$$

Similarly, the expected value and variance of the robust prediction y_{robust} can be estimated from the following equations:

$$\mathbf{E}[y_{robust}] = \hat{y}^* + \mathbf{E}[A_f], \tag{5.10}$$

and

$$\mathbf{Var}[y_{robust}] = \mathbf{Var}[A_f], \tag{5.11}$$

where $\mathbf{E}[A_f]$ and $\mathbf{Var}[A_f]$ represents the expected value and variance of the adjustment factor, and $\mathbf{E}[y_{robust}]$ and $\mathbf{Var}[y_{robust}]$ represents the expected value and variance of the robust estimate.

5.1.3 Confidence Interval for Robust Estimate

Next, to quantify the uncertainty in the robust prediction y_{robust} due to model uncertainty, confidence intervals are established. In particular, 5^{th} and 95^{th} percentiles derived from the robust prediction are used quantify the model uncertainty. In theory, this interval is likely to contain the true estimated value. As the model uncertainty is assumed to follow normal distribution, the confidence intervals (see [69]) are calculated from the following equations:

$$\overline{CI} = \mathbf{E}[y_{robust}] + 1.96\sqrt{\mathbf{Var}[y_{robust}]}, \tag{5.12}$$

$$\underline{CI} = \mathbf{E}[y_{robust}] - 1.96\sqrt{\mathbf{Var}[y_{robust}]}, \tag{5.13}$$

where $\overline{CI}$ and $\underline{CI}$ represents the upper and lower confidence intervals of the robust estimate.

5.1.3.1 Criterion for Selecting the Number of Identical Networks to be Constructed

Note that the number of M identical ANNs constructed Section 5.1.4 depends on a stopping criterion which ensures that convergence has been met in the iterative procedure given above. The criterion chosen for this procedure is given as follows:

$$argmin\{\mathbf{E}_x[y_{robust}] \approx y_{train} : \underline{y}_{robust} \leq y_{train} \leq \overline{y}_{robust}\} \tag{5.14}$$

5.1.4 Adaptive Procedure for Robust Artificial Neural Network Training

Here, the algorithm for implementing the above approach is reported in the following steps.

Step 1: Generate a training set $D_{train}(x, y)$ of input-output data set by sampling N_{train} independent input parameters values x_p, $p = 1, 2, ..., N_{train}$, and calculating the corresponding set of N_{train} output vectors $y_p = \mu_y(x_p)$ through the high fidelity model. Plain random sampling, such as Latin Hypercube Sampling (LHS) or other more sophisticated experimental design methods such as Sobol' sampling can be adopted to select the input vectors $\mathbf{x}_p, p = 1, 2, ..., N_{train}$.

Step 2: Construct a set $\mathcal{S}$ of $M > 1$ identical ANNs $f_m(\mathbf{x}, w_m^{\alpha*}), k = 1, 2, ...M$ (note that $f_m(\mathbf{x}, w_m^{\alpha*}) \equiv N_m$) on the basis of the entire data set D_{train} (step 1) to obtain fast-running surrogates of the non-linear deterministic function $\mu_y(x)$.

Step 3: Compute $P(N_m|D_{train})$ of $f_m(\mathbf{x}, w_m^{\alpha*})$.

Step 4: Use the ANNs $f_m(\mathbf{x}, w_m^{\alpha*})$ (step 2), in place of the expensive model, to provide a point estimate $\hat{Q}_m$ of the quantity Q, e.g., failure probability or sensitivity indices. Specifically, draw samples of N_T new input vectors $\mathbf{x}_r, r = 1, 2, ..., N_T$, from the corresponding probability distributions and feed into the ANNs $f_m(\mathbf{x}, w_m^{\alpha*})$. Thereafter, use the corresponding output vectors $\mathbf{y}_r = f(\mathbf{x}_r, w^{\alpha*}), r = 1, 2, ..., N_T$, to calculate the estimate $\hat{Q}_m$ for Q. Since the ANNs $f_m(\mathbf{x}, w_m^{\alpha*})$ can be evaluated quickly, this step is computationally cheap even if the number N_T model calls is very high (e.g., $N_T = 10^5 - 10^6$).

Step 5: Combine the estimates $\hat{Q}_m$ to provide a robust estimate $\hat{Q}_{robust}$ of the quantity of interest $\hat{Q}$.

Step 6: Calculate the two-sided robust confidence intervals $\overline{\hat{Q}}_{robust}$ and $\underline{\hat{Q}}_{robust}$ based on Equation 5.12 and 5.13 respectively. It is important to note that for a correct quantification of the confidence interval the estimate $\hat{Q}_m$ must be based on the same input and output vectors $\mathbf{x}_r$ and $\mathbf{y}_r, r = 1, 2, ..., N_T$

Step 7: Check if the stopping criterion based on Equation 5.14 is met. If met, proceed to the next step, else, return to Step 2 and generate $M + 1$ identical ANNs.

Step 8: End the algorithm. The statistic of interest is estimated at this step.

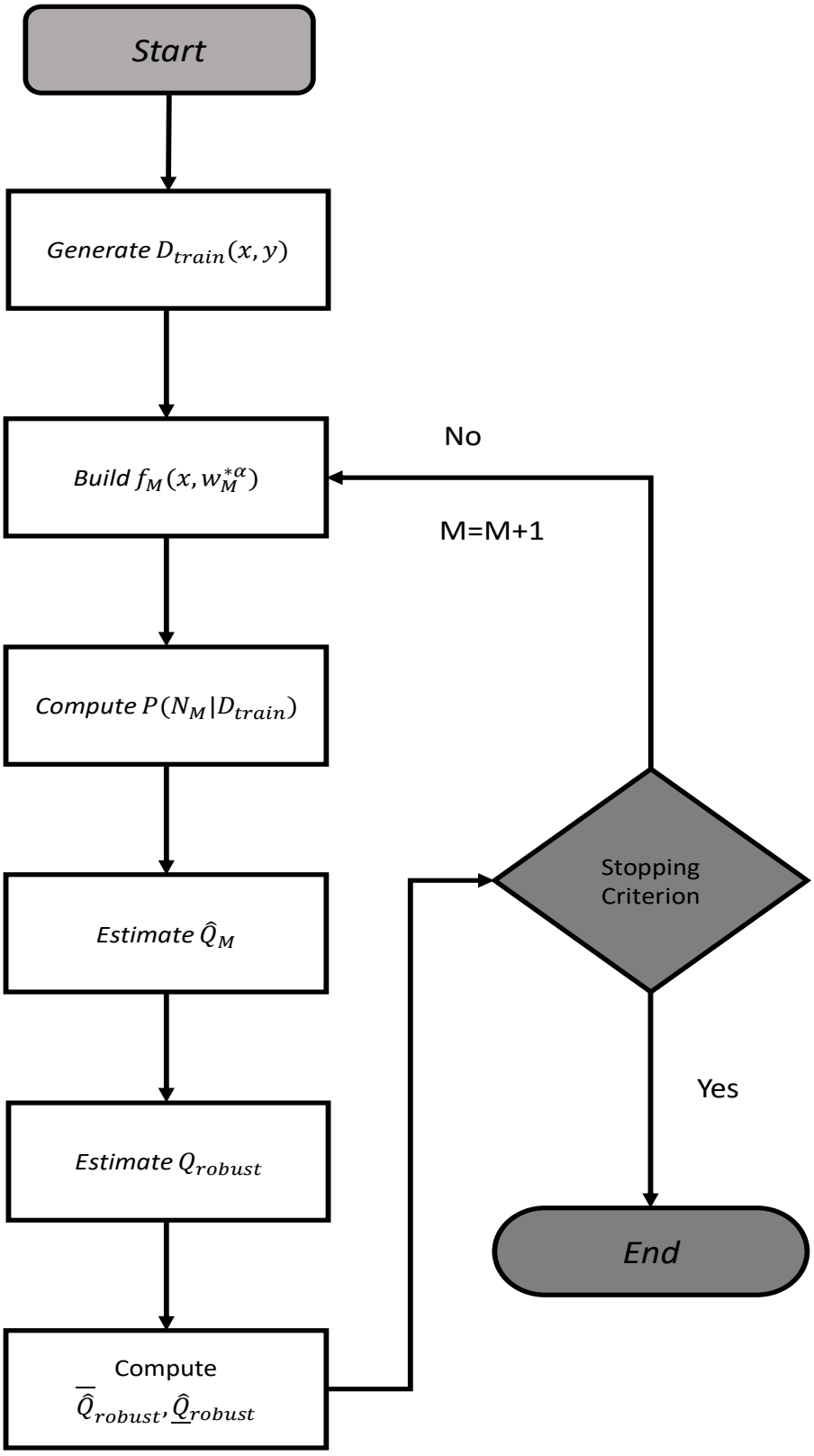


Figure 5.1: Flow Chart for Adaptive Robust ANN Algorithm

5.2 Case Study

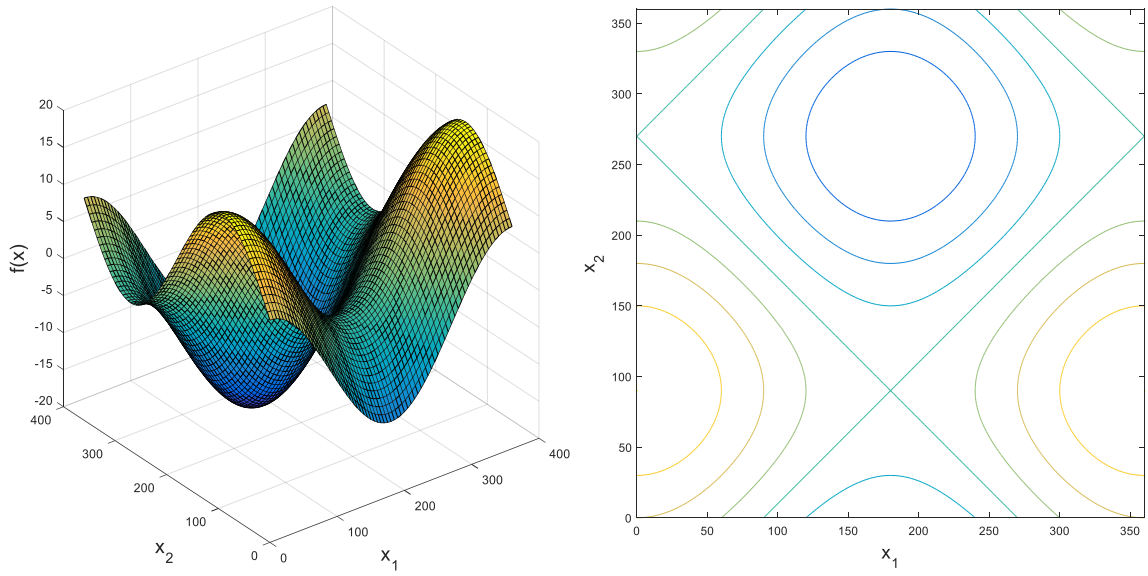
To demonstrate the applicability of the approach presented, two numerical examples are tested.

5.2.1 Case Study 1: The 2-D Non-Linear Function

The first example presents a 2-D non-linear limit state function. The limit state is given by the equation:

$$f(\mathbf{x}) = 10cos(x_1) + 10sin(x_2) \tag{5.15}$$

where the input parameters x_1 and x_2 are uniformly and independently distributed in the range of 0 and 360 degrees $\mathcal{U}(0, 360)$. In this example, the failure criteria of the limit state function is defined as $f(\mathbf{x}) \geq 15$. The surface and contour plot of the function is shown in Figure 5.2.



(a) Surface Plot (b) Contour Plot

Figure 5.2: Surface and Contour Plot of Limit State Function

Analysis

To implement the proposed approach, training samples $D_{train}(x, y)$ of size $N_{train} = 200$ have been generated from the limit state function via Latin hypercube sampling (LHS) design algorithm [22]. Thereafter, the proposed method has been followed, adopting $N_T = 10^4$ MC samples to estimate the failure probability $\hat{p}_{F_{robust}}$ and their corresponding confidence intervals $\underline{\hat{p}_{F_{robust}}}$ and $\overline{\hat{p}_{F_{robust}}}$.

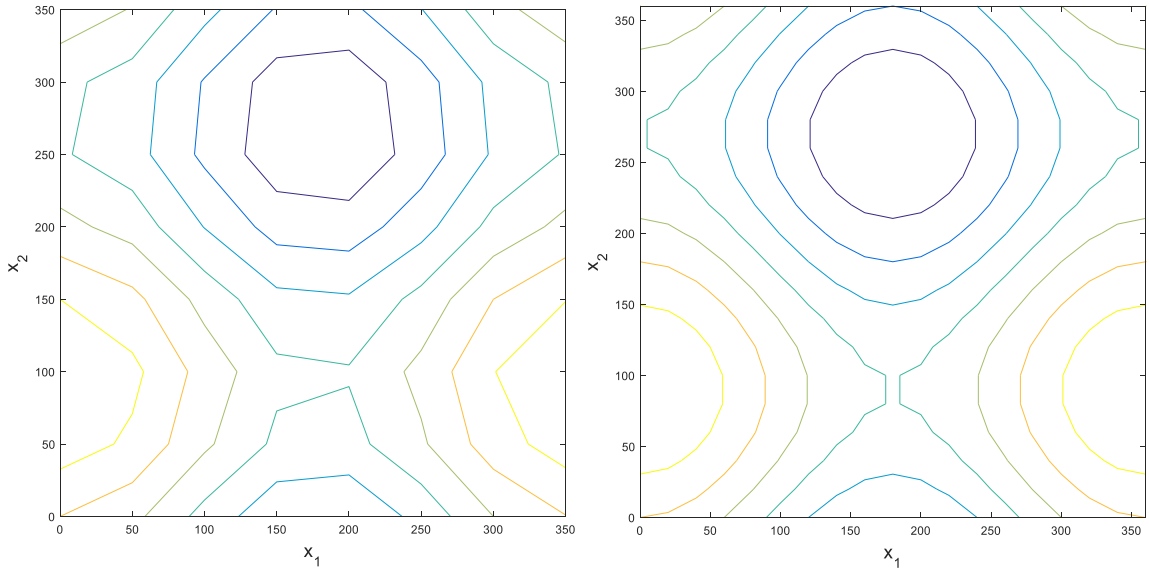
Results

Table 5.1 shows $\hat{p}_{F_{robust}}$, $\underline{\hat{p}_F}_{robust}$ and $\overline{\hat{p}_F}_{robust}$ obtained from the proposed approach after the stopping criterion was achieved after $M = 200$ iterations of the algorithm. The results have been compared to a reference solution obtained with $N = 10^5$ MC samples.

Table 5.1: Estimate of the Failure Probability Using Robust ANN

$\underline{\hat{Q}}_{robust}$	$\hat{Q}_{robust}$	$\overline{\hat{Q}}_{robust}$	$Reference\ \hat{p}_F$
0.075	0.085	0.094	0.087

Figure 5.3 shows the limit state obtained from different models. The real limit-state is compared to the limit state estimated by a classical ANN and the robust ANN derived from the proposed algorithm in this chapter. As seen in Figure 5.3(b), failure region (orange contour line) in the limit state plot from the robust ANN closely matches the failure region from the real limit state surface plot in Figure 5.2. However, the limit state plot from the classical ANN slightly underestimates the failure region (see Figure 5.3(c)). Clearly, using the robust ANN to estimate the failure probability $\hat{p}_F$ provide a more accurate estimate, when compared to the estimate obtained from the classic ANN.



(a) Classic ANN (b) Robust ANN

Figure 5.3: Limit State Surface Plot - Safety Function

Additionally, Table 5.2 shows the percentage of true value of the validation data that falls inside the confidence bounds computed using the algorithm presented in this

chapter. In the same table, error bounds have been estimated from 200 trials with classical ANN. From the results, 98% of the validation samples fall within the confidence intervals at $M = 200$ iteration. However, using 200 classic ANN to compute error bounds $\min[\hat{Q}]$ and $\max[\hat{Q}]$, 75.5% of the validation samples fall within the bounds. Thus, the bounds computed from the approach proposed in this paper is accurate in terms of encapsulating the true value to be predicted.

Table 5.2: Accuracy of Proposed Approach - 2-D Non-Linear Function

Approach	Accuracy	Computational Time	RMSE
Classic ANN (200 trials)	75.5%	0.029s	0.06
M=50	73%	0.109s	0.055
M=100	82%	0.238s	0.045
M=200	98%	0.357s	0.041
M=500	98%	0.488s	0.040

Robust Prediction Intervals Estimation

To assess the quality of the estimated robust confidence intervals $\underline{\hat{Q}}_{robust}, \overline{\hat{Q}}_{robust}$ the coverage probability and the interval width introduced in Chapter 4 is considered here. Such that the $PICP$ in this chapter is given as:

$$PICP = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} C_i \text{ with } \begin{cases} C_i = 1, & \text{if } Q_i \in [\underline{\hat{Q}}_{robust}, \overline{\hat{Q}}_{robust}] \\ C_i = 0, & \text{otherwise} \end{cases} \tag{5.16}$$

Similarly, the $NMPIW$ is given as:

$$NMPIW = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{\underline{\hat{Q}}_{robust} - \overline{\hat{Q}}_{robust}}{\hat{Q}_{robust}} \tag{5.17}$$

The objectives in this section is to produce the tightest robust intervals with the largest probability of containing the real value Q. Thus, the $PICP$ is maximised, while the $NMPIW$ is minimised.

Table 5.3: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (200 trials)	0.92	0.45
$M = 50$	0.90	0.40
$M = 100$	0.92	0.41
$M = 200$	0.94	0.39
$M = 500$	0.94	0.39

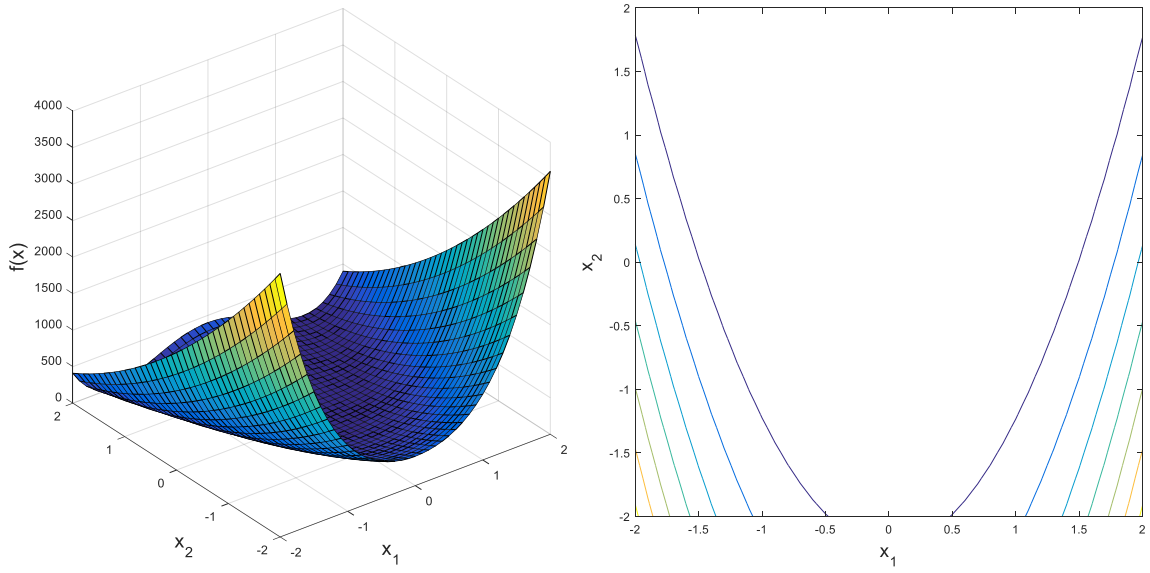
Similar to Table 4.2 in Chapter 4, Table 5.3 shows the $PICP$ and $NMPIW$ values obtained from different robust ANNs. As shown in Table 5.3, it can be seen that the best $PICP$ and $NMPIW$ values are obtained at $M = 200$. However, the $PICP$ and $NMPIW$ values estimated from the interval obtained from 200 classical ANNs are relatively lower. Thus, the approach proposed in this chapter is considered as a more robust approach.

5.2.2 Case Study 2: The Rosenbrock Function

The second example presents the Rosenbrock function. The Rosenbrock function has a two-dimensional input space and is defined by:

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \tag{5.18}$$

$x_i, i = 1, 2$ is modelled as a uniform random variable $\mathcal{U}(-2, 2)$. The response is non-monotone around the origin of the input space. Figure 5.4 shows the surface and contour plot of the Rosenbrock function. In this example, the failure criteria of the limit state function is defined as $f(\mathbf{x}) \geq 2000$.



(a) Surface Plot (b) Contour Plot

Figure 5.4: Surface and Contour Plot of Rosenbrock Function

Analysis Similar to the previous experimental settings, training samples $D_{train}(x, y)$ of size $N_{train} = 200$ have been generated. Then, the proposed method have been followed, adopting $N_T = 10^4$ MC samples to compute the failure probability $\hat{p}_{F_{robust}}$ and the corresponding confidence intervals $\underline{\hat{p}_{F_{robust}}}$ and $\overline{\hat{p}_{F_{robust}}}$.

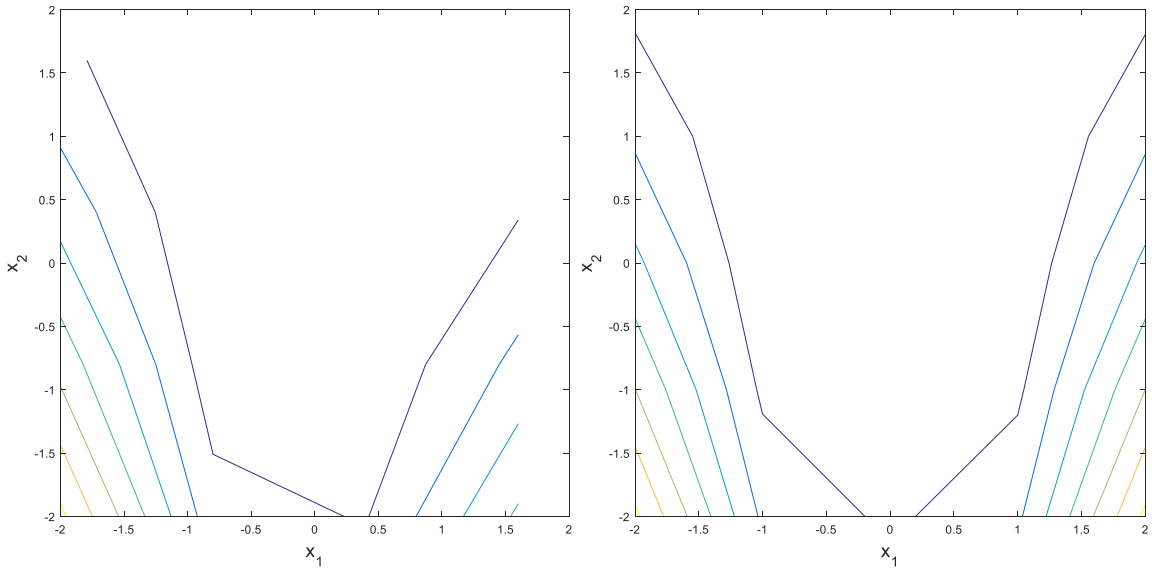
Results

The proposed algorithm converged after $M = 150$ iterations, and the results are shown in Table 5.4. Furthermore, the results have been compared to a reference solution obtained from the real model using $N = 10^5$ MC samples.

Table 5.4: Estimate of the Failure Probability Using Robust ANN

$\underline{\hat{Q}}_{robust}$	$\hat{Q}_{robust}$	$\overline{\hat{Q}}_{robust}$	$Reference\ \hat{p}_F$
0.058	0.065	0.077	0.063

The visual representation of the response surface predicted by classical ANN and the robust ANN is shown in Figure 5.5(a) and (b) respectively.



(a) Surface Plot from Conventional ANN (b) Surface Plot from Robust ANN

Figure 5.5: Surface and Contour Plot of Rosenbrock Function

From Figure 5.5(a), the global maximum at the bottom right of the figure is not detected by the classic ANN. Contrarily, the robust ANN (Figure 5.5(b)) captures the global maximum of the input space more accurately than the classic ANN. On the other hand, although the surface plot obtained from the robust ANN do not exactly match the real surface plot, any quantity of interest that is to be predicted via the robust ANN will be significantly more accurate than the prediction made by the classic ANN.

Table 5.5: Accuracy of Proposed Approach - Rosenbrock Function

Approach	Accuracy	Computational Time	RMSE
Classic ANN (150 trials)	63.5%	0.029s	0.07
M=20	73%	0.139s	0.065
M=50	82%	0.268s	0.052
M=100	95%	0.387s	0.038
M=150	98%	0.478s	0.035
M=200	98%	0.762s	0.03

Similar to Table 5.2, Table 5.5 shows the percentage of true value of the validation data that falls inside the confidence bounds computed for different number of M identical ANNs. From this result, the confidence bound computed from the proposed approach using $M = 150$ ANNs captures a higher percentage of validation data when compared with the error bounds obtained from 150 ANNs. The $PICP$ and $NMPIW$ values of these ANNs are shown in Table 5.6.

Table 5.6: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (150 trials)	0.93	0.42
M=20	0.93	0.43
M=50	0.93	0.41
M=100	0.94	0.42
M=150	0.95	0.40
M=200	0.95	0.40

On the basis of the results obtained, the proposed approach can be considered more effective in regression problems, while quantifying the uncertainty associated to the results.

5.3 Chapter Summary

In this chapter, a novel approach has been proposed to quantify the uncertainties introduced in an ANN due to the random initialization of the network weights by the training algorithm. The proposed approach combines Bayesian statistics and a model averaging technique into a unified framework that can be solved in parallel for more complex ANN architectures. It has been shown from several examples in this chapter that the proposed approach in this chapter significantly improves the accuracy of the

prediction. Furthermore, it has also been shown that the confidence bounds derived from this approach encapsulates the predicted value of interest. Hence, the proposed approach in this chapter is an efficient meta-modelling technique in terms of predictive capability. The downside of this approach is the computational time required, due to the large number of ANNs to be trained. However, as we are currently in an era of efficient computational resources, parallelization strategies can be adopted to reduce the wall-clock time.

Chapter 6

Robust Surrogate Models - Model Structure and Random Model Parameters

This chapter extends the approach proposed in Chapter 5 by the inclusion of an additional approach that takes into account the effect of uncertainty due the structure of the ANN (i.e. number of hidden layers, number of neurons neurons, and type of activation function). The optimal number of training samples required to train the ANN is also incorporated in this current approach. Thereafter, the same analytical functions used in Chapter 5 are tested with the approach in order to compare the robustness of this current approach and that proposed in Chapter 5.

6.1 Background to Problem

Although ANNs are great tools for mapping non-liner input-output relationships of complex critical systems, there is no guarantee that they will perform well for a given task. In fact, the performance of an ANN is related to the structure adopted. Consequently, it is of common practice within the ANN community to train a variety of ANNs with different structures, thereafter, the best ANN is selected on the basis of performance on an independent test set (i.e. data set not seen by the ANN). However, as this technique is based on trial and error heuristic approach, the best performing ANN may never be found. Thus, the results are uncertain. Additionally, training a large set of ANN with different structures and selecting one based on a performance test is waste of computational resource. Furthermore, an important issue that isn't usually considered when training a network is the adequate partitioning of the training, validation, and test data presented to the ANN. Often, it is of common

practice to use 70% of the available data to train the network, and the remaining 30% data for validation and testing purpose (usually split equally). However, following this standard procedure is not always suitable for the problem at hand. As a matter of fact, for some regression task it is desirable that all the available data be presented to the network in order to minimize over-fitting, as over-fitting wouldn't be an issue when a comprehensive training data set is presented to the ANN (i.e. all the possible input-output pairs available). Furthermore, the approach developed for the issues discussed in Chapter 5 are incorporated into this current chapter. Thus, a generalized approach that takes into account model structure and random model parameters is proposed in this chapter.

6.2 Proposed Approach

The underlining idea of the proposed approach in this chapter is to obtain a set $\mathcal{V}$ of optimal ANN architectures, where each respective error function is trained to unique local minimum. Thereafter, adopting the approach proposed in Chapter 5. The current approach proposed in this chapter address issues overlooked in Chapter 5 such as:

- The optimal model structure (i.e. number of layer, neurons, and activation function) that maximizes the predictability of the ANN.
- The optimal number of training sample size needed to train the ANN.

6.2.1 Formulation of Optimization Problem for Optimal ANN Architecture and Training Sample Size Selection

The first step of the approach is to formulate a multi-objective optimization problem that selects various optimal model structures. By means of mathematics, the problem is formulated as:

$$\begin{aligned} &\min && (f_1(z), f_2(z), ..., f_k(z)) \\ &\text{subject to} && g_c(z) \leq 0, q = 1, 2, ..., q, \end{aligned} \tag{6.1}$$

where the integer $k \geq 2$ represents the number of objective function, and $z \in X$ represents the feasible set of design vectors. This feasible set is defined by some set of constraints.

Objective Function

Here, the objective function is defined as $f : X \to \mathbb{R}^k$, $f(z) = (f_1(z), f_2(z), ..., f_k(z))^T$. An element $z^* \in X$ is the feasible solution. It should be noted that in multi-objective

optimization setup, there is no typical feasible solution that minimizes all the objective functions $(f_1(z), f_2(z), ..., f_k(z))$ simultaneously. Hence, attention is paid to Pareto curves that cannot be computed efficiently in many cases. Furthermore, using the weighted sum method [70], the objective functions $(f_1(z), f_2(z), ..., f_k(z))$ are combined into one single-objective scalar function. Thus, the objective function can be reformulated as:

$$f_{objective} = \min \sum_{i=1}^{k} \gamma_i \cdot f_i(z), \tag{6.2}$$

where the the sum of weights γ_i must obey the relationship:

$$\sum_{i=1}^{k} \gamma_i = 1, \gamma_i > 0, i = 1, 2, ..., k. \tag{6.3}$$

k has been chosen to be equal to 2, hence, the first objective function $f_1(z)$ is the prediction square error PSE [71] expressed mathematically as :

$$PSE = \xi_{train} + 2\sigma^2 \frac{p}{N}, \tag{6.4}$$

where p denotes the total number of weight parameters in the ANN dependant on the architecture, and σ^2 the estimate of the true error variance which is not dependent on the network architecture being considered and is approximated by [72]:

$$\hat{\sigma}^2 = \frac{1}{N-p} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2. \tag{6.5}$$

Thereafter, the second objective function $f_2(z)$ defined, is used to identify the number of training samples N_{train}. The optimal number of training samples is defined as the sample size that minimizes the square difference between training error and the validation error at the point where the training stops (early stopping). Mathematically, $f_2(z)$ is defined as:

$$E_{diff} = (E_{train} - E_{validation})^2. \tag{6.6}$$

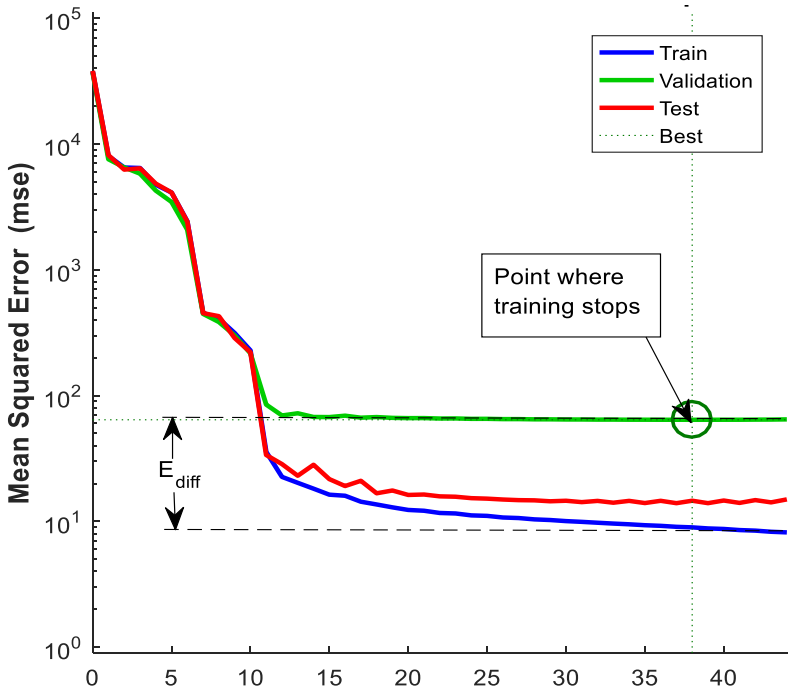


Figure 6.1: Region Where E_{diff} is Computed

Design Variables

Here, the design variables $z \in X$ used in the multi-objective optimization problem are:

- Number of hidden layers.
- Number of nodes in the hidden layer.
- Activation function.
- Number of training samples.

Constraints

Subsequently, two constraints have been chosen for this problem. The first constraint $g_1(z)$ guarantees that there should be at least one hidden layer present in a network architecture, composing of neurons that can vary within the interval 1-32. Further, the second constraint $g_2(z)$, ensures that the set of activation functions to be tested are limited to, Sigmoid activation function:

$$f(x) = \frac{2}{1 + \exp^{2x}} - 1, \tag{6.7}$$

Linear activation function;

$$f(x) = x, \tag{6.8}$$

and Gaussian activation function;

$$f(x) = \exp^{-x^2}. \tag{6.9}$$

Thus, the set $\mathcal{V}$ of optimal solution consists of ANNs with different combinations of hidden layer architecture, activation function and, number of training samples.

6.2.1.1 Searching Optimal ANN Solutions with Evolutionary Algorithm

Thereafter, to efficiently search the solution space for good solutions, Genetic Algorithm (GA) [54] has being adopted due to its stochastic nature and ability to locate global optimal solutions for complex optimization problems. GA are optimization methods aiming at finding the global optimum of one (or more) real objective function(s) of one or more decision variables, possibly subject to various linear or non linear constraints. Their main properties are that the search is conducted (1) using a population of multiple solution points or candidates, (2) using operations inspired by the evolution of species, such as breeding and genetic mutation, (3) using probabilistic operations, (4) using only information on the objective function and not on its derivatives. The chromosomes in a GA population take the form of bit strings. In particular, each chromosome represents a candidate solutions to the problem at hand. The algorithm processes populations of chromosomes, replacing one population with another using a fitness function that assigns a fitness value to each chromosome in the current population being processed.

6.2.1.2 Encoding the Chromosome

Specifically, an indirect encoding scheme have been employed to encode the chromosome, such that each chromosome carries information about the number of hidden layers and number of neurons within a hidden layers, activation function and number of training samples. In particular, the solution is coded in a $25 - bit$ binary chromosome. The first, second, and third group of $5 - bit$ correspond to the number of neurons in the first, second and third hidden layer, respectively. Consequently, due to the binary coding scheme, the number of neurons in each hidden layer falls within the interval $\{0, 31\}$. Thus, the chromosome contains information for both the number of hidden layers and the number of neurons in each layer. Furthermore, from the remaining $10 - bit$, the information about the activation function used is stored in $2 - bit$, where each activation function is referred to as numbers 1,2, and 3, where 1 represents hyperbolic tangent, 2 linear, and 3 Gaussian. Finally, the remaining $8 - bit$ are used to store information regarding the percentage of training samples used to

train the ANN. It is obvious that by increasing the number of available bits, any ANN architecture can be created, so this encoding scheme ensures general applicability. For the purpose of this work, the limits of 3 hidden layers and 31 neurons in each hidden layer, are imposed for computational reasons.

6.2.1.3 Procedures Taken to Search for Optimal Network Architectures

In this section, the procedures taken in the optimization of the ANN architecture are given in the following steps:

Step 1: First, an initial number of **P** chromosomes are generated by GA (i.e. **P** is chosen by the analyst), where each chromosome contain a network structure encoded in bit strings as described in Section 6.2.1.2.

Step 2: Next, **P** ANNs are initialized is subsequently trained with the training algorithm along with early stopping algorithm. For this purpose, the available data must be divided into three subsets, namely the training, validation (for avoiding overfitting) and testing subset (for performance evaluation). It should be noted that the number of training data set is encoded in each respective chromosome.

Step 3: Thereafter, the fitness function value of each ANN architecture is evaluated.

Step 4: Then, repeat the following steps until **P** offspring have been produced:

- a Select a pair of parent chromosomes from the current population, the probability of selection p_s being an increasing function of fitness value. Selection is done with replacement, meaning that the same chromosome can be selected more than once to become a parent.
- b With probability of cross over p_c, cross over the pair at a randomly chosen point (chosen with uniform probability) to form two offspring. If no crossover takes place, form two offspring that are exact copies of their respective parents. Note that the crossover rate is defined to be the probability that two parents will cross over in a single point.
- c Mutate the two offspring at each locus with probability p_m (the mutation probability or mutation rate), and place the resulting chromosomes in the new population. If **P** is odd, one new population member can be discarded at random.

Step 5: Replace the current population with the new population.

Step 6: Go to step 3.

Importantly, it should be noted that each iteration of the above procedure is called a generation. Furthermore, the entire set of generations is called a run. Finally, at the end of running the algorithm, there are often one or more highly fit chromosomes in the population.

6.2.1.4 Ensemble of Optimal Networks

Subsequently, a vector $\mathcal{V}$ consisting of fit chromosomes representing various ANN architectures that have been trained to different local minima is produced from the optimization set-up.

6.2.2 Bayesian Model Selection for Matrix Consisting of Identical ANNs

Next, the optimal ANNs in $\mathcal{V}$ are trained repeatedly in order to take account of the random initialization of the weight of the ANNs in $\mathcal{V}$. Thus, leading to the production of matrix $\mathbf{A}_{m,d}$ of size $M \times D$ consisting of ANNs $N_{m,d}$. Matrix $\mathbf{A}_{m,d}$ is given as:

$$\mathbf{A}_{m,d} = \begin{bmatrix} N_{1,1} & N_{1,d} & \dots & N_{1,D} \\ N_{m,1} & N_{m,d} & \dots & N_{m,D} \\ \vdots & \vdots & \ddots & \vdots \\ N_{M,1} & N_{M,d} & \dots & N_{M,D} \end{bmatrix}$$

Such that each column of $\mathbf{A}$ contains a vector of unique identical ANNs architecture. The idea behind the formation of matrix $\mathbf{A}_{m,d}$ is to take into consideration the different sources of uncertainties arising from (1) the ANN architecture and, (2) the random initialization of the weight parameter. Thus, given matrix $\mathbf{A}_{m,d}$, Bayesian statistics can be used to infer the posterior probability of the m^{th} ANN in d^{th} column as:

$$P(N_{m,d}|D_{train}) = \frac{P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})P(N_{k,l})}{\sum_{m=1}^{M} P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})P(N_{m,d})}, \tag{6.10}$$

where $P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})$ is the likelihood of training data $D_{train}(\mathbf{x},\mathbf{y})$ for the $N_{m,d}$ ANN, and $P(N_{m,d})$ is the prior probability of $N_{m,d}$, which is the ANN probability evaluated before observing training data $D_{train}(\mathbf{x},\mathbf{y})$. The prior ANN probability $P(N_{m,d})$ can be specified depending on the existing prior knowledge about the credibility of ANN $N_{m,d}$, or it can be given as a uniform probability, $P(N_{m,d}) = 1/M$, if no additional information is provided. The advantage of assigning uniform prior probability to $P(N_{m,d})$ is that the difficulty of estimating the prior probability numerically

is avoided. The likelihood $P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})$ may be thought of as the probability of observing the training data $D_{train}(\mathbf{x},\mathbf{y})$ under $N_{m,d}$ ANN. It supplies a relative measure of how well the $N_{m,d}$ ANN is supported by the training data $D_{train}(\mathbf{x},\mathbf{y})$. Since the denominator in Eq.(6.10) is common for all the ANNs, the posterior ANN probability is proportional to prior probability and the likelihood. The likelihood of each ANN is evaluated by measuring the degree of agreement between the training data $D_{train}(\mathbf{y})$ and the response $\hat{y}^z_{m,d}$ for each ANN. Hence, a probabilistic relationship between training data $D_{train}(\mathbf{x},\mathbf{y})$ and ANN predictions $\hat{y}^z_{m,d}$ involving uncertainty can be described. Typically, the bias function and noise are included as parts of the probabilistic relationship to match ANN predictions with training data. The bias function captures the discrepancies between the expensive model responses and predictions made by the ANN. The noise is usually assumed to be independent and identically distributed normal random variable with a mean of zero [64]. Furthermore, various authors, [65–67] have used the Bayesian statistical method to quantify the uncertainty in the bias function modelled as a Gaussian process. In their works, a mathematical formulation that combines bias function associated with the ANN and noise from training data is utilized to describe the probabilistic relationship between the training data $D_{train}(\mathbf{x},\mathbf{y})$ and ANN predictions $\hat{y}^z_{m,d}$. The mathematical formulation of this probabilistic relationship is given by the following:

$$D_{train}(\mathbf{y}) = \hat{y}^z_{m,d} - \varepsilon^z_{m,d}, \tag{6.11}$$

where $\varepsilon^z_{m,d}$ is a random variable that covers both bias associated with the ANN prediction $\hat{y}^z_{m,d}$ and the noise in the response training data $D_{train}(\mathbf{y})$. $\varepsilon^z_{m,d}$ is assumed to be an independent identically distributed random variable with a mean $\mu_{m,d}$ of zero. The use of $\varepsilon^z_{m,d}$ with zero mean does not shift ANN prediction $\hat{y}^z_{m,d}$. This reflects the fact that $\hat{y}^z_{m,d}$ is the most probable prediction value for the ANN. The bias function is not included as a separate term in the probabilistic relationship. This is due to the fact that introducing a separate bias function results in shifting the prediction $\hat{y}^z_{m,d}$ of the ANN from the initially predicted value. Next, the likelihood $P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})$ of training data $D_{train}(\mathbf{x},\mathbf{y})$ for ANN $N_{m,d}$ is evaluated by observing where the training data points $D_{train}(\mathbf{y})$ are located in the distribution of $\hat{y}^z_{m,d}$ estimated by $N_{m,d}$. The procedures to estimate the distribution $P(\hat{y}|N_{m,d})$ of $N_{m,d}$ and the likelihood $P(D_{train}(\mathbf{x},\mathbf{y})|N_{m,d})$ is given. First, the uncertainty in errors of predictions $\hat{y}^z_{m,d}$ made by $N_{m,d}$ is quantified by introducing an assumption that the prediction errors are independent and identically distributed normal random variable

with a mean $\mu_{m,d}$ of zero. The error of the prediction of network $N_{m,d}$ is represented by the following:

$$\varepsilon^z_{m,d} = D_{train}(y^z) - \hat{y}^z_{m,d}, \varepsilon^z_{m,d} \sim N(0, \sigma^2_{m,d}), z = 1, 2, ..., N, \tag{6.12}$$

where $D_{train}(y^z)$ is the z^{th} training response output data, $\hat{y^z}$ the prediction of the training data made by $N_{m,d}$, $\sigma^2_{m,d}$ is the variance of prediction error $\varepsilon^z_{m,d}$, and N the number of samples in the training data. The prediction error $\varepsilon^z_{m,d}$ measured is considered to be a random sample from a normal distribution with a mean $\mu_{m,d}$ of zero and variance $\sigma^2_{m,d}$. Using the principle of maximum likelihood estimation (MLE) (see [68]), the variance $\sigma^2_{m,d}$ for $N_{m,d}$ can be estimated as:

$$\sigma^2_{m,d} = \frac{1}{N}\sum_{z=1}^{N}(\varepsilon^z_{m,d})^2. \tag{6.13}$$

Furthermore, the predictive distribution $P(\hat{y}|N_{m,d})$ of response $\hat{y}_{m,d}$ under ANN $N_{m,d}$ is created by including the prediction error obtained in the previous step into the prediction of $\hat{y}_{m,d}$ made by $N_{m,d}$. This predictive distribution is defined by the following equation:

$$P(\hat{y}_{m,d}|N_{m,d}) = D_{train}(\mathbf{y}) + \varepsilon^z_{m,d}. \tag{6.14}$$

Lastly, assuming that the residuals between the training data $D_{train}(\mathbf{x}, \mathbf{y})$ and ANN $N_{m,d}$ output $\hat{y}_{m,d}$ are normally and independently distributed with a mean of zero and constant variance $\sigma^2_{m,d}$, the likelihood function $P(D_{train}(\mathbf{x}, \mathbf{y})|N_{m,d})$ can be expressed as:

$$P(D_{train}(\mathbf{x}, \mathbf{y})|N_{m,d}) \approx \frac{1}{\sqrt{2\pi\sigma^2_{m,d}}}\frac{1}{N}\sum_{z=1}^{N}\exp\{\frac{-[y^z - \hat{y}^z_{m,d}]^2}{2\sigma^2_{m,d}}\}. \tag{6.15}$$

6.2.2.1 Robust Prediction from Artificial Neural Networks

Thereafter, to obtain a robust prediction $y^d_{robust}, d = 1, 2, ..., D$ from the robust network (i.e. set comprising of identical networks in the d^{th} column of $\mathbf{A}_{m,d}$), the prediction made by all the identical networks in the d^{th} column of $\mathbf{A}_{m,d}$ are combined using the adjustment factor approach introduced in Chapter 5. Hence, the robust prediction of the ANNs in $\mathbf{A}_{m,d}$ is expressed as:

$$y^d_{robust} = \hat{y}^{d*} + A^d_f, d = 1, 2, ..., D, m = 1, 2, ..., M, \tag{6.16}$$

where $\hat{y}^{d*}$ represents the point estimate of the best ANN in the d^{th} column characterised by the highest posterior probability, and y^d_{robust} represent the robust prediction

from the identical networks in the d^{th} column of $\mathbf{A}_{m,d}$. Since the adjustment factor A_f^d is assumed to be normally distributed, the expected value $\mathbf{E}[\cdot]$ and variance $\mathbf{Var}[\cdot]$ of the adjustment factor A_f^d is given by the following:

$$\mathbf{E}[A_f^d] = \sum_{m=1}^{M} P(N_{m,d}|D_{train})(\hat{y}^{m,d} - \hat{y}^{d*}), \tag{6.17}$$

for $d = 1, 2, ..., D$ and $m = 1, 2, ..., M$

$$\mathbf{Var}[A_f^d] = \sum_{m=1}^{M} P(N_{m,d}|D_{train})(\hat{y}^{m,d} - \mathbf{E}[y_{robust}^d])^2. \tag{6.18}$$

Similarly, the expected value and variance of the robust prediction y_{robust}^d can be estimated from the following:

$$\mathbf{E}[y_{robust}^d] = \hat{y}^{d*} + \mathbf{E}[A_f^d], \tag{6.19}$$

$$\mathbf{Var}[y_{robust}^d] = \mathbf{Var}[A_f^d], \tag{6.20}$$

where $\mathbf{E}[A_f^d]$ and $\mathbf{Var}[A_f^d]$ represents the expected value and variance of the adjustment factor, and $\mathbf{E}[y_{robust}^d]$ and $\mathbf{Var}[y_{robust}^d]$ represents the expected value and variance of the robust estimate.

6.2.2.2 Confidence Interval for Robust Neural Network Prediction

Next, to quantify the uncertainty in the robust prediction y_{robust}^d due to model uncertainty, confidence intervals are established. In particular, the 5^{th} and 95^{th} percentiles of the robust prediction are used quantify the model uncertainty. In theory, this interval is likely to contain the true estimated value. As the model uncertainty is assumed to be distributed normally, the confidence intervals of each respective ANN are given as:

$$\overline{y}_{robust}^d = \mathbf{E}[y_{robust}^d] + a^* \sqrt{\mathbf{Var}[y_{robust}^d]}, \tag{6.21}$$

and

$$\underline{y}_{robust}^d = \mathbf{E}[y_{robust}^d] - a^* \sqrt{\mathbf{Var}[y_{robust}^d]}. \tag{6.22}$$

where $\overline{y}_{robust}^d$ and $\underline{y}_{robust}^d$ represents the upper and lower confidence intervals of the robust estimate from N_{robust}^d and a^* represents the upper critical value of the Gaussian distribution quantifying the model uncertainty.

6.2.3 Model Averaging for the Ensemble of Robust Neural Networks

Since the true regression function which we seek to approximate is denoted by $\mu(x)$, the mapping function of each robust network N^d_{robust} in $\mathcal{V}$ can be defined as the true function with noise given by the relationship:

$$N^d_{robust}(x) = \mu(x) + \epsilon_l, d = 1, 2, ..., D. \tag{6.23}$$

The average sum of square error for network $N^d_{robust}(x), d = 1, 2, ..., D$ can be written as:

$$E_d = \mathbf{E}[\{N^d_{robust}(x) - \mu(x)\}^2] = \mathbf{E}[\epsilon^2_d]. \tag{6.24}$$

The expectation of the error square $\mathbf{E}[\epsilon^2_d]$ corresponds to an integration over x which is weighted by the joint density $\Omega(x)$ of x such that:

$$\mathbf{E}[\epsilon^2_d] \equiv \int \epsilon^2_d(x)\Omega(x). \tag{6.25}$$

The average error made by each robust network $N^d_{robust}(x)$ is given by:

$$E_{ave} = \frac{1}{D}\sum_{d=1}^{D} E_d = \frac{1}{D}\sum_{d=1}^{D} \mathbf{E}[\epsilon^2_d]. \tag{6.26}$$

The prediction from the ensemble of robust networks can be written as:

$$y_{essem} = \frac{1}{D}\sum_{d=1}^{D} N^d_{robust}(x). \tag{6.27}$$

The error due to the ensemble of robust networks can be written as:

$$E_{essem} = \mathbf{E}[(\frac{1}{D}\sum_{d=1}^{D} N^d_{robust}(x) - \mu(x))^2] = \mathbf{E}[(\frac{1}{D}\sum_{d=1}^{D} \epsilon_d)^2]. \tag{6.28}$$

In addition, making an assumptions that the error $\epsilon_d, d = 1, 2, ..., D$ of each robust network $N^d_{robust}, d = 1, 2, ..., D$ has a zero mean, and uncorrelated such that $\mathbf{E}[\epsilon_d\epsilon_q] - \mathbf{E}[\epsilon_d]\mathbf{E}[\epsilon_q] = 0$ where $d \neq q$. Hence, the ensemble of robust networks is related to the average error by the relationship:

$$E_{essem} = \frac{1}{D^2}\sum_{d=1}^{D} \mathbf{E}[\epsilon^2_d] = \frac{1}{D}E_{ave}. \tag{6.29}$$

6.2.3.1 Combination of the Robust Networks Confidence Intervals

Furthermore, to combine the confidence intervals $\{\overline{y}^d_{robust}, \underline{y}^d_{robust}\}, d = 1, 2, ..., D$ from the ensemble of robust ANNs $N^d_{robust}, d = 1, 2, ..., D$ in order to effectively capture the uncertainty in the prediction, vertex method [73] has been adopted. With this method, the intervals of the ensemble of robust ANNs $\{\underline{y}^d_{robust}, \overline{y}^d_{robust}\}$ are propagated from a sum average function. The ordinates of the vertices $c_{d,j}$ are elements matrix $\mathbf{B}_{d,j}$ of size $D \times 2^D, j = 1, 2, ..., 2^D$:

$$\mathbf{B}_{d,j} = \begin{bmatrix} c_{1,1} & c_{1,j} & \dots & c_{1,2^D} \\ c_{d,1} & c_{d,j} & \dots & c_{d,2^D} \\ \vdots & \vdots & \dots & \vdots \\ c_{D,1} & c_{D,j} & \dots & c_{D,2^D} \end{bmatrix}$$

The values in each column of $\mathbf{B}_{d,j}$ represent ordinates used to evaluate the sum average function. Thus, the robust intervals can be calculated from the following equations:

$$\underline{y}_{robust} = \min_j(f(c_{d,j})), d = 1, 2, ..., D, j = 1, 2, ..., 2^D, \tag{6.30}$$

and

$$\overline{y}_{robust} = \max_j(f(c_{d,j})), d = 1, 2, ..., D, j = 1, 2, ..., 2^D. \tag{6.31}$$

6.2.3.2 Criterion for Selecting the Number of Identical Networks to be Constructed

Note that the number of identical ANNs M to be constructed depends on a stopping criterion which ensures that convergence has been met in the iterative procedure given above. The linear convex optimization problem that needs to be satisfied in order for the algorithm to stop reads:

$$argmin\{\mathbf{E}_x[y_{robust}] \approx y_{train} : \underline{y}_{robust} \leq y_{train} \leq \overline{y}_{robust}\}. \tag{6.32}$$

6.2.3.3 Adaptive Procedure for Optimized Robust ANN Training

Here, the algorithm for the above approach iteratively are reported in the following steps.

Step 1: Generate a set $D_{train}(x, y)$ of input-output data set by sampling N_{train} independent input parameters values x_p, $p = 1, 2, ..., N_{train}$, and calculating the corresponding set of N_{train} output vectors $y_p = \mu_y(x_p)$ through the high fidelity model. Experimental design algorithm, such as Latin Hypercube Sampling (LHS) or a more sophisticated algorithm such as Sobol' sequence sampling [21] can be adopted to select the input vectors $x_p, p = 1, 2, ..., N_{train}$.

Step 2: Construct a set $\mathcal{V}$ of optimal ANNs from the multi-objective optimization discussed above with GA.

Step 3: Generate $M > 1$ identical ANNs, and construct a matrix $\mathbf{A}_{m,d}$, where the size of $\mathbf{A}_{m,d}$ is $M \times D$.

Step 4: Compute the posterior probability $P(N_{m,d}|D_{train})$ of all the ANNs in $\mathbf{A}_{m,d}$.

Step 5: Use the ANNs in $\mathbf{A}_{m,d}$, in place of the expensive model, to provide a point estimate $\hat{Q}_{m,d}$ of the quantity Q, e.g., failure probability or sensitivity indices. Specifically, draw a sample of N_T new input vectors $x_r, r = 1, 2, ..., N_T$, from the corresponding probability distributions and feed into the ANNs in $\mathbf{A}_{m,d}$ with them; then, use the corresponding output vectors $y_r = f(x_r, w^*), r = 1, 2, ..., N_T$, to calculate the estimate $\hat{Q}_{m,d}$ for Q. Since the ANNs in $\mathbf{A}_{m,d}$ can be evaluated quickly, this step is computationally cheap even if the number N_T of model evaluations is very high (e.g., $N_T = 10^5 - 10^6$).

Step 6: Combine the estimates $\hat{Q}_{m,d}$ to provide a robust estimate $\hat{Q}^d_{robust}$ of the quantity of interest $\hat{Q}$.

Step 7: Calculate the two-sided robust confidence intervals $\overline{\hat{Q}}^d_{robust}$ and $\underline{\hat{Q}}^d_{robust}$. It is important to note that for a correct quantification of the confidence interval the estimate $\hat{Q}_{m,d}$ must be based on the same input and output vectors x_r and $y_r, r = 1, 2, ..., N_T$

Step 8: Check if the stopping criterion based on Equation 6.32 is met. If met, proceed to Step 9, else, return to Step 3 and construct $M = M + 1$ identical ANNs.

Step 9: End the algorithm. The statistic of interest is estimated at this step.

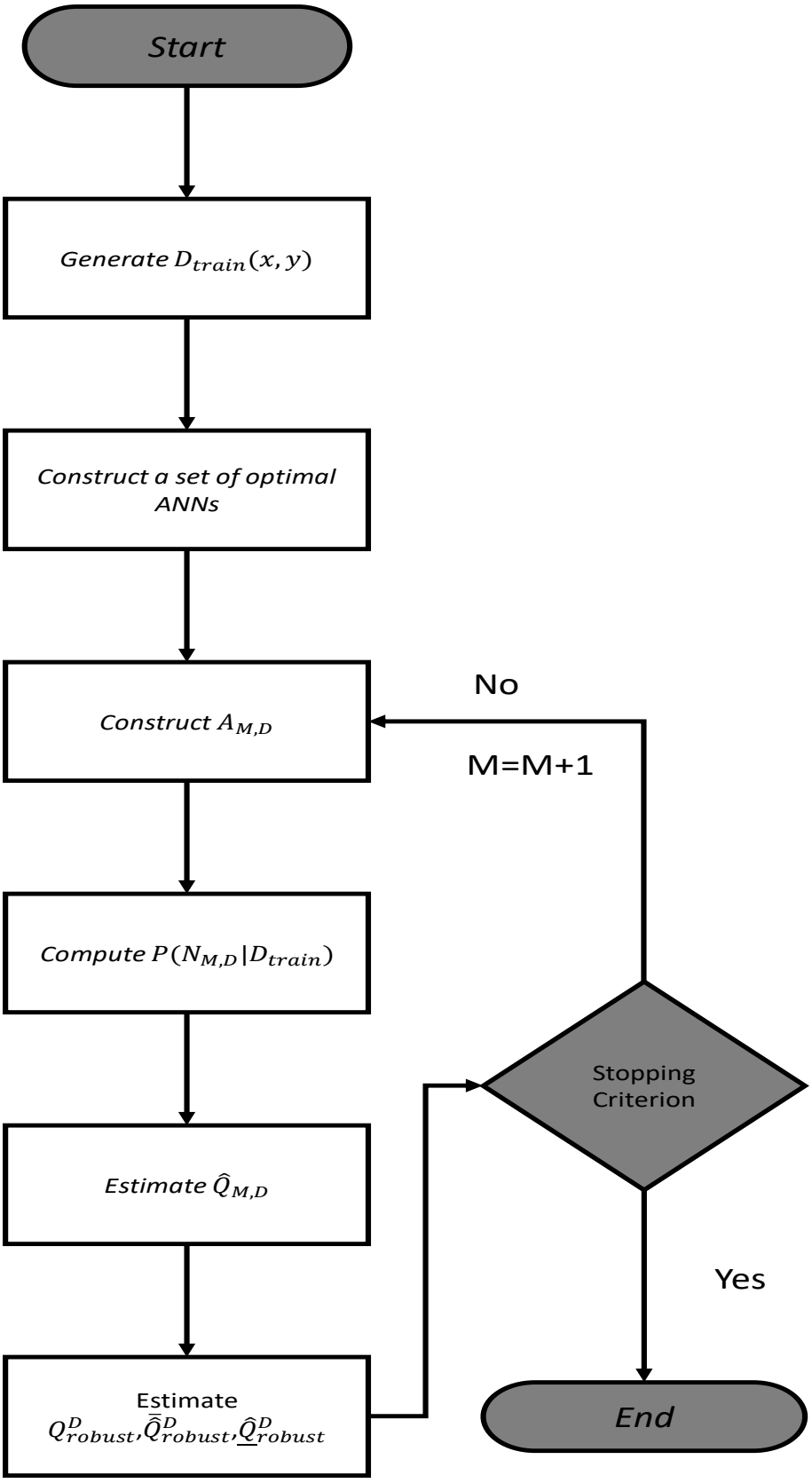


Figure 6.2: Flow Chart for Adaptive Optimized Robust ANN Algorithm

6.3 Case Study

6.3.1 Case Study 1

Here, to compare the proposed approach in this chapter with the approach in Chapter 5, the same case study adopted in Chapter 5 have been adopted here.

6.3.1.1 Analysis

Subsequently, the same experimental settings followed in Chapter 5 have been adopted here. In particular, LHS technique have been used to draw the training samples. Furthermore, a population $P = 50$ chromosomes has been chosen for the multi-objective optimization problem.

6.3.1.2 Results

From 100 generations of the GA algorithm in both numerical examples, the optimal ANN architectures discovered from both analyses are shown in Table 6.1 and 6.2.

Hidden Layers	*Activation Function*	*Neurons*	*Training Samples*
1	*Gaussian*	{2,16,1}	70%
1	*Hyperbolic Tangent*	{2,12,15,1}	80%
2	*Gaussian*	{2,22,30,1}	80%
2	*Linear*	{2,16,18,1}	80%

Table 6.1: Optimal ANN Architectures from GA Optimization - Safety Function

Hidden Layers	*Activation Function*	*Neurons*	$\%of D_{train}$
1	*Gaussian*	{2,18,1}	70%
2	*Hyperbolic Tangent*	{2,10,8,1}	75%
2	*Gaussian*	{2,15,19,1}	85%

Table 6.2: Optimal ANN Architectures from GA Optimization - Resenbrock Function

Thereafter, in both respective examples, the adaptive algorithm converged at $M = 55$ iterations for the safety function problem and at $M = 60$ iterations for Rosenbrock function, while observing the same experimental settings in Chapter 5.

Table 6.3 and 6.4 shows the estimate of the failure probability obtained from these analyses.

Table 6.3: Estimate of the Failure Probability Using Robust ANN

$\underline{\hat{Q}}_{robust}$	$\hat{Q}_{robust}$	$\overline{\hat{Q}}_{robust}$	$Reference\ \hat{p}_F$
0.080	0.086	0.089	0.087

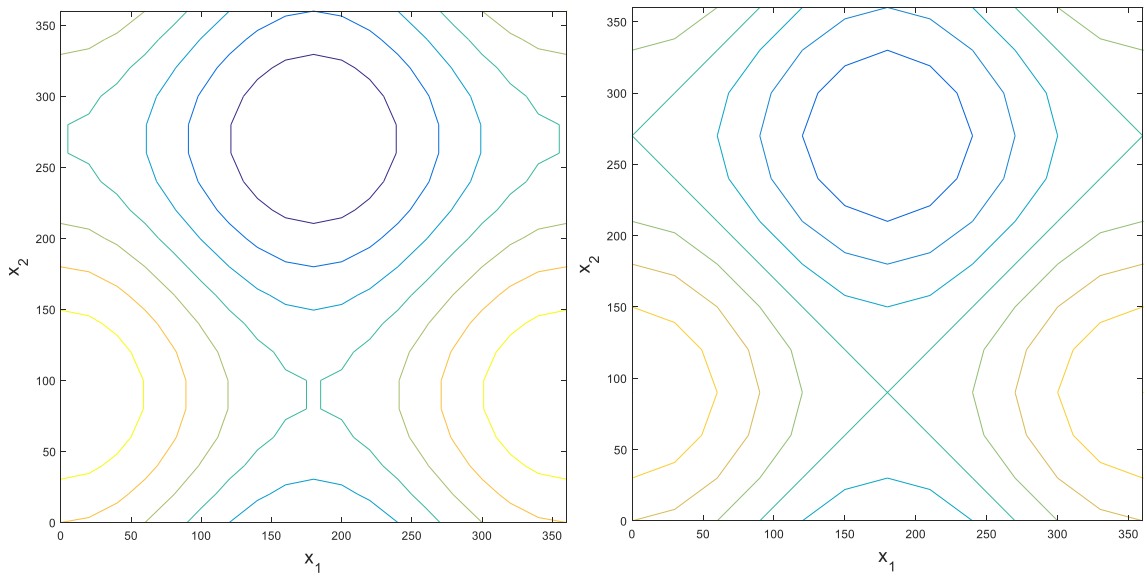
Table 6.4: Estimate of the Failure Probability Using Robust ANN

$\underline{\hat{Q}}_{robust}$	$\hat{Q}_{robust}$	$\overline{\hat{Q}}_{robust}$	$Reference\ \hat{p}_F$
0.061	0.064	0.074	0.063

From the results given in the tables, it is shown that the result obtained from the approach in this current chapter closely matches the reference solution with higher a higher degree of accuracy when compared to the results obtained from Chapter 5 approach. Moreover, the confidence bounds obtained from this current approach is tighter and still encapsulates the true value of the failure probability. Thus, on the basis of the results obtained, the approach proposed in this chapter can be considered more effective than the approach in Chapter 5 in the estimation of the failure probability while quantifying the uncertainty associated to the results because they provide more accurate estimates are closer to the true values and precise confidence intervals are narrower.

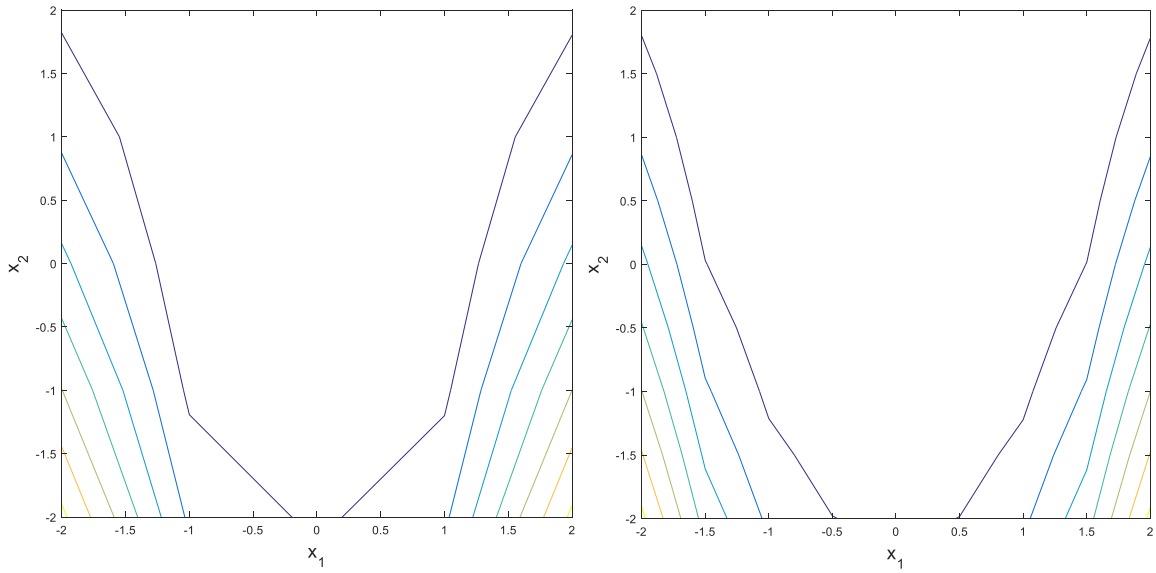
Visual Comparison of Surface Plot for the different approaches

Additionally, Figure 6.3 and 6.4 show the contour plot of the modelling approach adopted in Chapter 5 and the approach proposed in this chapter. From the results, it can be seen that the approach proposed in this chapter estimates the response surface of both functions with a higher degree of accuracy compared to ANN modelling approach presented in Chapter 5. The reason for the higher accuracy in the results from the approach in this chapter is due to the fact that optimal ANN architectures have been utilized in the analysis, thus, improving the generalization properties.



(a) ANN in Chapter 5 (b) ANN in this Chapter

Figure 6.3: Response Surface Plot - Safety Function



(a) ANN in Chapter 5 (b) ANN in this Chapter

Figure 6.4: Response Surface Plot - Resenbrock Function

Performance Evaluation

Here, a comparison of the performance of both modelling approaches in terms of the percentage of validation data that falls within their respective confidence intervals are given in Tables 6.5 and 6.7 respectively. Furthermore, the $PICP$ and $NMPIW$

values of the robust ANNs used in the respective case studies are shown in Table 6.6 and 6.8.

Table 6.5: Accuracy of Method for 2D-Non-Linear Function

Approach	Accuracy	Computational Time	RMSE
Classic ANN (55 trials)	52%	0.029s	0.06
M=10	70%	0.109s	0.035
M=20	75%	0.238s	0.038
M=30	81.5%	0.357s	0.038
M=55	85%	0.488s	0.038
M=60	85%	0.792s	0.038

Table 6.6: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (55 trials)	0.92	0.45
$M = 10$	0.90	0.40
$M = 20$	0.92	0.41
$M = 30$	0.94	0.39
$M = 55$	0.95	0.38
$M = 60$	0.95	0.38

Table 6.7: Accuracy of Method for Rosenbrock Function

Approach	Accuracy	Computational Time	RMSE
Classic ANN (60 trials)	50%	0.029s	0.07
$M = 10$	70%	0.139s	0.055
$M = 20$	75%	0.268s	0.045
$M = 30$	78%	0.387s	0.035
$M = 60$	85%	0.478s	0.025
$M = 70$	85%	0.762s	0.020

Table 6.8: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Classic ANN (60 trials)	0.93	0.42
$M = 10$	0.93	0.43
$M = 20$	0.93	0.41
$M = 30$	0.94	0.42
$M = 60$	0.96	0.39
$M = 70$	0.96	0.39

As shown in, Tables 6.5 and 6.7 the results obtained from the proposed approach in this chapter outperforms that of Chapter 5. Furthermore, the $PICP$ and $NMPIW$ values obtained in Table 6.6 and 6.8 further confirms the accuracy of the proposed approach in this chapter. On the basis of these results, the proposed approach is more suitable for quantifying surrogate models uncertainties and improving their accuracy.

6.3.2 Case Study 2

To compare the proposed framework with another state-of-the-art surrogate modelling technique, a more realistic engineering example is considered. The engineering problem considered in this chapter is a borehole function, which simulate the water flow through a borehole. This benchmark model has been discussed in [74]. The model consists of an eight-dimensional input vector $x = (r_w, r, T_u, H_u, T_l, H_l, L, K_w)^T$ [74]:

$$h(x) = \frac{2\pi T_u (H_u - H_l)}{ln(r/r_w)(1 + \frac{2LT_u}{ln(r/r_w)r_w^2 K_w} + \frac{T_u}{T_l})}, \tag{6.33}$$

where $h(x)$ is the fluid water flow measured in m^3/year, r_w is the radius of the borehole, r the radius of influence, T_u the transmissivity of the upper aquifer, H_u the potentiometric head of the upper aquifer, T_l the transmissivity of the lower aquifer, H_l the potentiometric head of the lower aquifer, L the length of the borehole, and K_w the hydraulic conductivity of the soil. The variability in the input vector are modelled as independent random variables whose properties are summarized in Table 6.9. For the log-normal distribution, the parameters are the mean and standard deviation of the natural logarithm of the variable. For the other variables, they describe the range of uniform distributions.

Table 6.9: Borehole Model Definition of the Probabilistic Model of the Input Parameters

Parameters	Units	Distribution	Parameters
r_w	$[m]$	Uniform	[0.05, 0.15]
r	$[m]$	Log-normal	[7.71, 1.0056]
T_u	$[m^2/year]$	Uniform	[63070, 115600]
H_u	$[m]$	Uniform	[990, 1110]
T_l	$[m^2/year]$	Uniform	[63.1, 116]
H_l	$[m]$	Uniform	[700, 820]
L	$[m]$	Uniform	[1120, 1680]
K_w	$[m^2/year]$	Uniform	[9855, 12045]

For the purpose of this example, it is been assumed that the borehole function is expensive to run, thus, the developed framework in this chapter (i.e. robust ANN) and a Kriging model are adopted and compared in terms of their predictive capability and width of the confidence bounds.

6.3.2.1 Analysis

For the experimental settings, training data of size $N = 200$ have been generated from the expensive model and used to train a Kriging and a robust ANN in parallel. In particular, a two hidden layer ANN architecture of [8, 8, 5, 1] have been discovered by the GA, while in the Kriging model, an exponential kernel function have been used. For the robust ANN training procedure proposed, $M = 20$ have been chosen to for computational reasons.

6.3.2.2 Results

The statistics of interest is the water flow $h(x)$ corresponding to combination of different input values. Figure 6.5 and 6.6 shows the prediction from the developed approach in this chapter and a Kriging model respectively. Comparing both figures (Figure 6.5 and 6.6), it is clearly shown that the prediction made by the robust ANN matches the target data with better precision compared to the Kriging model. In addition, the width of the confidence intervals from the robust ANN is tighter than that from the robust ANN.

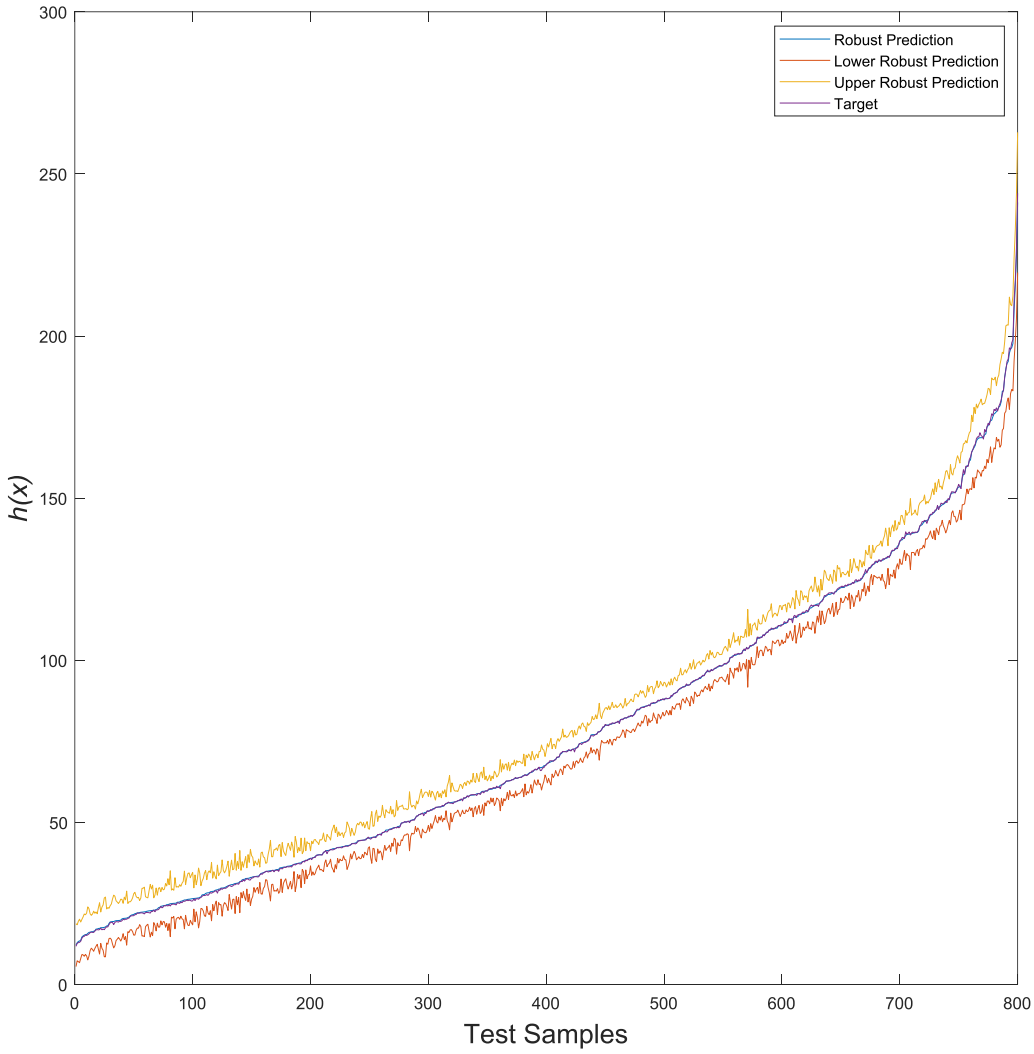


Figure 6.5: Robust ANN Prediction

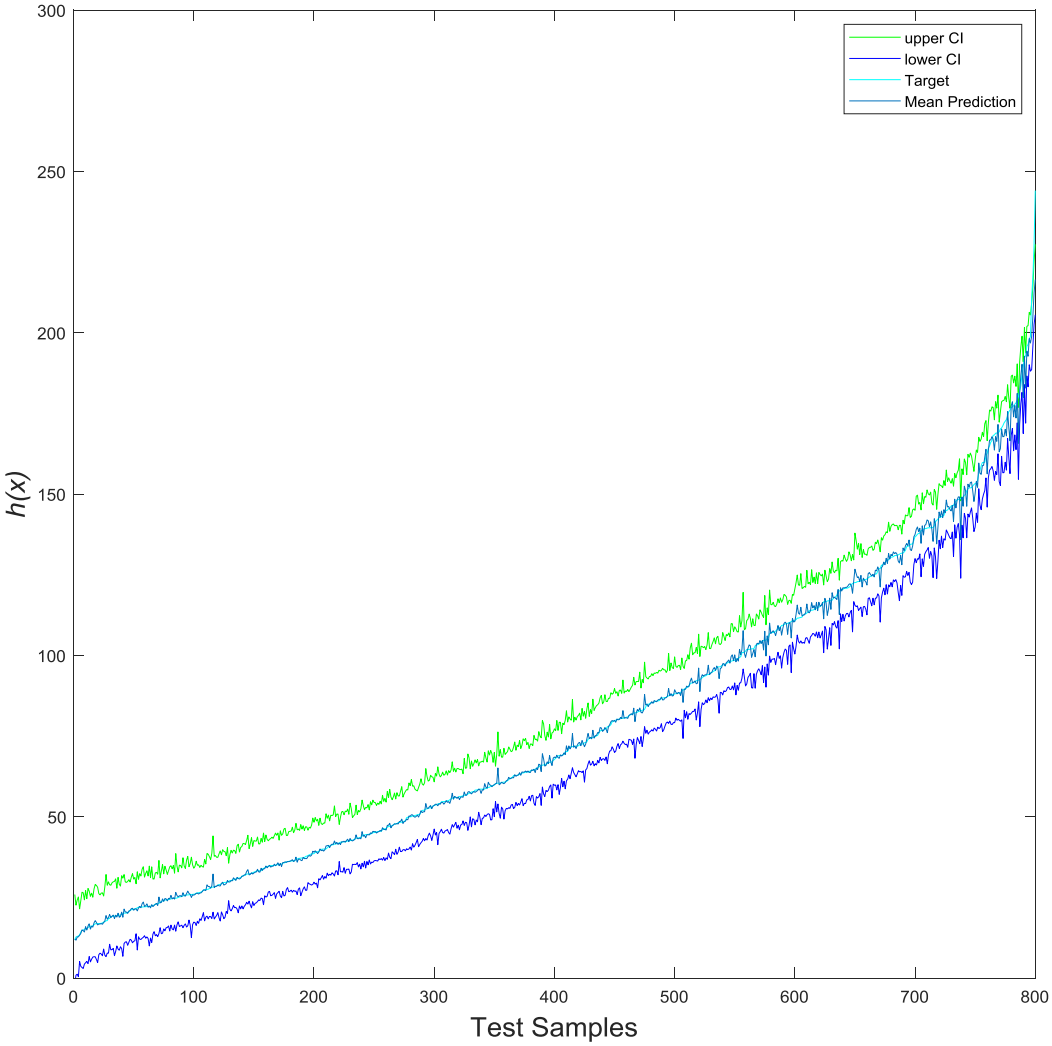


Figure 6.6: Kriging Prediction

Table 6.10: Prediction Intervals Performance Comparison Between the Proposed Approach and Classical Approach

Approach	$PICP$	$NMPIW$
Kriging Model	0.93	0.42
Robust ANN	0.95	0.40

Furthermore, the $PICP$ and $NMPIW$ value for the two surrogate models are shown in Table 6.10. Similarly, Table 6.10, shows that the robust ANN performs better in terms of the coverage probability and width. On the basis of these obtained results, we can adopt the proposed approach in this chapter to a more complex realistic engineering problem.

6.4 Chapter Summary

In this chapter, a novel approach for the quantification of ANN uncertainty as a result of the model structure and random initialization of the weight parameters in the model is proposed. The approach in this chapter extends the approach proposed in Chapter 5 by including optimization and an interval propagation approach. The approach in this chapter have been compared with the approach in Chapter 5 on the basis of simple numerical examples. Particularly, the intervals obtained from this approach tend to be smaller than the intervals obtained in Chapter 5. Although, the percentage of validation data that falls inside the interval calculated from this approach is lower, the RMSE is relatively smaller. Furthermore, the width of the confidence bounds obtained in this approach is relatively smaller than that of Chapter 5. Thus, we can conclude that from this approach the model uncertainty is reduced due to the tighter confidence bounds. On the other hand, adopting this approach requires a lot of computational resource, as a huge number of surrogate models are required to be trained. However, parallelization strategies can be adopted to reduce the computational cost.

Chapter 7

Uncertainty Quantification of the Site Ion eXchange Plant Using Robust Artificial Neural Networks

Nuclear decommissioning is a prime example of an uncertainty quantification task due to large uncertainties involved in dealing with nuclear materials. Thus, in this chapter, a nuclear decommissioning problem is set up and solved with the approaches developed in Chapter 4 and 5. In particular, the approaches are adopted for reliability and sensitivity analysis problem focusing on a process simulation model of a UK nuclear effluent treatment plant. This model has been extensively validated against plant and experimental data and used to support the UK effluent discharge strategy.

7.1 Overview

Currently, at Sellafield Nuclear decommissioning site, programmed activities are onging to make a significant advance in the retrieval and decommissioning its legacy Ponds and silos. Particularly, the programme is scheduled for at least another 100 years, with an estimated cost in the tens of billions of pounds [1]. Subsequently, this wide ranging cost estimate reflects the uncertainties and contingencies within the programme due to the considerable technical, environmental and operational challenges. On the other hand, the wastes stored at the Sellafield Site have been generated over an extended time scale (1950s present). Interestingly, in the case of the legacy wastes generated up to the early 1980s there is some uncertainty about the condition of these wastes and the facilities they are stored in. This was due to poor record keeping at the time, and continued limited access due to high radiation hazards which poses challenges for sampling and detailed characterisation. Consequently, there is an un-

certainty of the future effluent arising that will be generated as part of the retrieval and decommissioning operations in terms of the volume generated and the chemical compositions. To mitigate this uncertainty, a key facility to support current decommissioning operations being conducted at the Sellafield is the Site Ion eXchange Plant (SIXEP) [1]. A schematic of the working process of the SIXEP is illustrated in Figure 7.1.

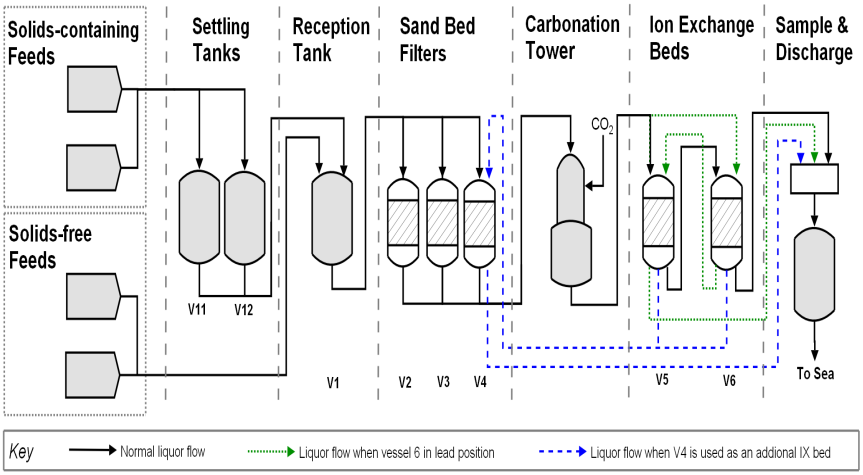


Figure 7.1: SIXEP Schematic [1]

From Figure 7.1, the feeds into the SIXEP contain particulate materials, and a number of soluble radioactive isotopes which are predominantly, ^{137}Cs and ^{90}Sr. These soluble radioactive species are removed from the liquid effluent using an ion exchange media loaded in 2 ion exchange beds which operates in series (one lead bed and one lag bed). The lead bed is replaced with fresh media when it is exhausted, and the bed that previously operated in the lag position is promoted to the lead position. The filtration and carbonation steps are present to protect the eXchange beds and have a secondary benefit of removing actinides. Furthermore, to ensure continued removal of these two key radioactive isotopes, the plant is routinely operated on the basis of feeds meeting a set of Conditions for Acceptance (CfA). These CfA define the feed envelop in terms of the acceptable concentrations of inactive species which affect the efficiency of the process.

7.1.1 Computational Model of the SIXEP

Generally speaking, the use of a modelling tool (SIXEP model) is relied upon to understand and optimise the retrieval and decommissioning programmes. This is because the model can provide:

- A way to interrogate what process variables need to be known. This will inform future monitoring requirements and to define and underpin the feed and operating envelope
- An underpinning of technical risks because models help to illustrate the consequences of operating outside of existing feed and /or operating envelopes.
- An up to date description of the technical processes that will in the future facilitate knowledge management.

Currently, the SIXEP model is being used to test new feed compositions to prove assurance that the plant can continue to operate effectively, i.e. ensuring the discharges of ^{137}Cs and ^{90}Sr are kept within the required limit. It should be noted that the SIXEP model evolved initially from a simple dynamic process model of the ion exchange columns, designed to replicate how small changes to feed activity would impact on effluent leaving the plant. Thus, throughout the SIXEP model evolution and development, the focus has always been on trying to understand plant behaviour, not simply to replicate it. Importantly, the SIXEP model has been implemented in the gPROMS modelling code, supplied by Process Systems Enterprise Limited. The gPROMS software allows a model of a plant or a process to be built from a collection of component models, linked together through connection streams. Therefore, the behaviour of the individual components can be defined using sets of time dependent differential and algebraic equations that have been designed to represent the chemical and physical processes that occur therewith.

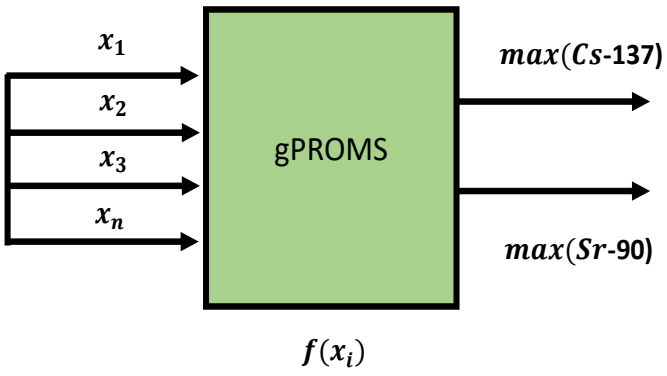


Figure 7.2: Black-Box Schematic of SIXEP

Conversely, there is uncertainty of the future feeds composition arising from the Sellafield site, leading to variability in the activity levels of ^{137}Cs and ^{90}Sr and other soluble species that affect the removal of these isotopes. This variability can lead to

undesirable consequences (i.e. the discharges of the two afore-mentioned radionuclides exceeds their desired levels). Hence, we wish to evaluate if it is practical to incorporate this uncertainty into studies when using the SIXEP model to assess the risk involved, and identify the parameters that contribute significantly to this variability. It is to be noted that the uncertainty considered to affect the plant feeds are aleatory (i.e. random) in nature. Thus, the consideration of these type of uncertainty in the SIXEP model leads to defining of a state vector $\mathbf{x}$ of 18 inputs of the SIXEP model $\mathbf{x} = x_n : n = 1, 2, .., 18$, which are characterized by the probability distributions given in Table 7.1. These uncertain inputs map out two output quantities of interest defined by the vector $y = y_z : z = 1, 2$. The outputs of interest of the model represents the maximum concentration of ^{137}Cs and ^{90}Sr respectively after approximately 540 days. Figure 7.3 shows a deterministic simulation from the SIXEP model using a set of representative mean values specified in Table 7.1.

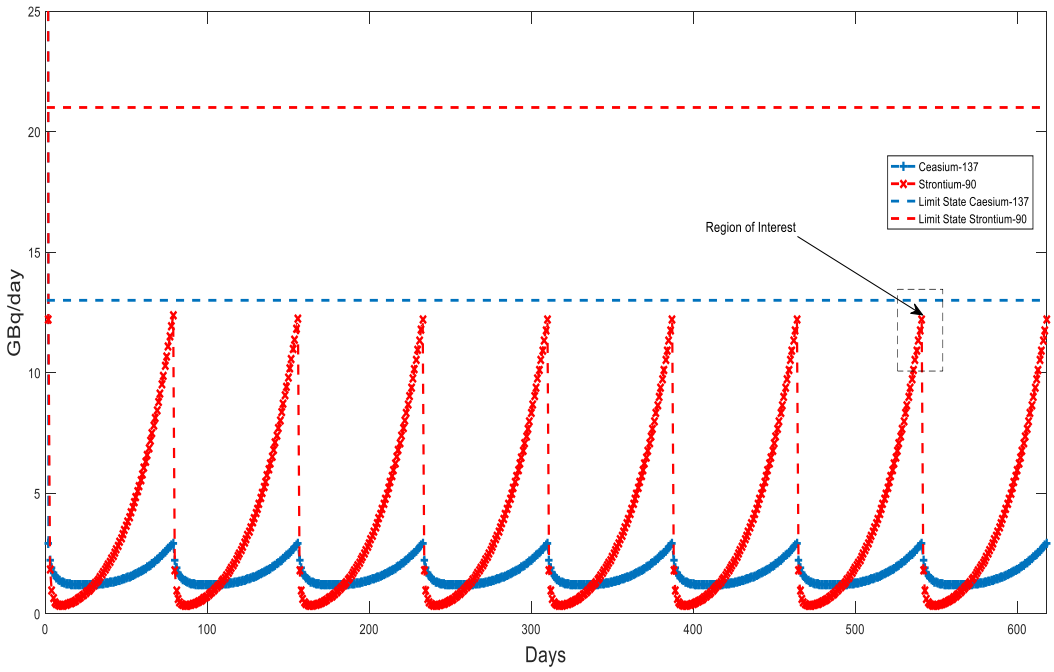


Figure 7.3: Deterministic Simulation from the SIXEP Model

The results from the deterministic evaluation of the SIXEP model shown in Figure 7.3 illustrates the activity levels of ^{137}Cs and ^{90}Sr with respect to time. From Figure 7.3, it can be see that there is a rise and drop in the activity levels of the ^{137}Cs and ^{90}Sr caused by the ion exchange bed change cycle. In this simulation, an ion exchange bed change occurs every 77 days (i.e. every 11 weeks). When a new ion exchange bed comes on-line, the activity discharges are low, and as the ion exchange media becomes

saturated, activity breaks through the bed and thus produces a rising discharge profile which drops again following the next bed change (shown as the peaks in Figure 7.3). Thereafter, propagating the input parameter uncertainties via MC simulation through the SIXEP model gives rise to variability in the maximum concentration of ^{137}Cs and ^{90}Sr at this point. It should be noted that the simulation shown in Figure 7.3, the SIXEP model has been started from a saved state representing steady state operation with the mean parameter values. When the parameter values are changed using the MC, it will take the model up to a maximum of 8 ion exchange bed change cycles to reach a new steady state, hence the scalar quantities of interest (maximum activity level of ^{137}Cs and ^{90}Sr) are taken at the end of 8 bed change cycles.

7.1.1.1 Modelling the Uncertainty in the Model Input Parameters

Here, the uncertainty affecting the inputs of the SIXEP are modelled by random variables (RV) described in Chapter 2. A RV is specified by selecting a desired distribution from an available pool and setting the relevant parameters. Alternatively, the distribution parameters can be inferred from available data by means of maximum likelihood procedures described in the literature of this thesis. Table 7.1 shows the random variables used to characterize the input parameter uncertainties. Note that the distribution parameters θ have been estimated via likelihood means from data collected such that:

$$\mathcal{L}(\mu, \sigma|\chi) = \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} exp[-\frac{1}{2}(\frac{\chi^{(i)} - \mu}{\sigma})^2] \tag{7.1}$$

7.2 Reliability Analysis of the SIXEP Using Robust Artificial Neural Networks

Definition of Failure Criteria

Here, the black-box schematic of the SIXEP model depicted in Figure 7.2 is considered failed when the concentration of ^{137}Cs and ^{90}Sr exceeds the limit state of 12GBq/day and 20 GBq/day respectively (see Figure 7.3) specified by the Environmental Agency (EA). Thus, the limit state of the SIXEP model is defined as:

$$F = \{f_{^{137}\mathrm{Cs}}(x) \geq 12\} \cup \{f_{^{90}\mathrm{Sr}}(x) \geq 20\} \tag{7.2}$$

Subsequently, exceeding these threshold values are expected to increase the level of radiation contamination in the Irish sea. Thus, it is necessary to quantify the

Table 7.1: Random Variables used to Represent the Uncertainties of the SIXEP Feed Inputs

$ParameterID$	$Distribution$	$\mu(\theta)$	$\sigma(\theta)$
1	Normal	5.00E+02	6.60E+02
2	Normal	3.90E+04	3.70E+04
3	Normal	1.05E+03	3.59E+05
4	Normal	3.00E+01	2.00E+01
5	Normal	4.60E-05	3.60E-05
6	Normal	6.13E-03	1.83E-03
7	Normal	1.59E-05	1.28E-05
8	Normal	9.40E-06	1.05E-05
9	Normal	1.59E+05	7.10E+04
10	Normal	4.50E+01	4.90E+01
11	Normal	2.00E+03	6.20E+02
12	Normal	3.30E+01	3.90E+01
13	Normal	1.40E+00	3.00E+00
14	Normal	3.84E-06	1.22E-05
15	Normal	3.50E-03	2.82E-04
16	Normal	3.20E-06	3.28E-06
17	Normal	2.38E-06	2.93E-06
18	Normal	2.00E+06	2.79E+05

probability $\hat{p}_F$ of this event. Furthermore, adopting the simulation based approach to estimating the failure probability $\hat{p}_F$ requires a repeated number of model evaluations. However, the SIXEP model is relatively expensive. In particular, a single model evaluation of the model requires 1200 seconds, hence it becomes infeasible to compute a robust estimate of the failure probability and the sensitivity indices. Thus, the ANN modelling approach introduced in Chapter 4 and 5 have been adopted for this study.

7.2.1 Analysis

Training data D_{train} of size $N_{train} = 800$ have been generated from gPROMS via LHS experimental design technique. Furthermore, bootstrap and identical ANNs with architecture of $\{18, 7, 2\}$ have been constructed based on the algorithms proposed in Chapter 4 and 5 respectively. Thereafter, the algorithm proposed in Chapter 4 converged after $B = 300$ iterations, while the algorithm proposed in Chapter 5 converged after $M = 100$ iterations, while adopting $N = 10^4$ MC samples to compute $\hat{p}_F$.

7.2.2 Results

Failure Probability Estimation

Figures 7.4 and 7.5 visualizes several iterative stages of the algorithms proposed in Chapter 4 and 5 respectively. The red lines in Figure 7.4 denotes the BBC estimate of $\hat{p}_F$, while the black lines represent the lower and upper BBC. From Figure 7.4, as the number of bootstrap ANNs B are increased, the width of the intervals gets tighter, until the stopping condition defined is met. Similarly, in Figure 7.5, the distribution quantifying the uncertainty in the estimate of $\hat{p}_F$ begins to narrow down as the number M is increased.

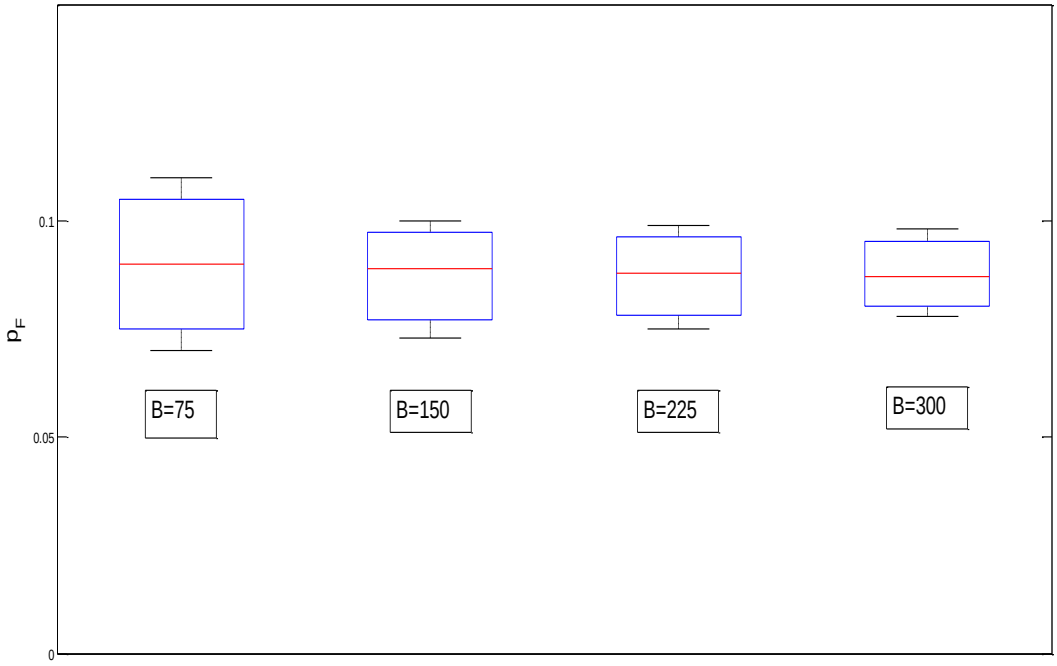
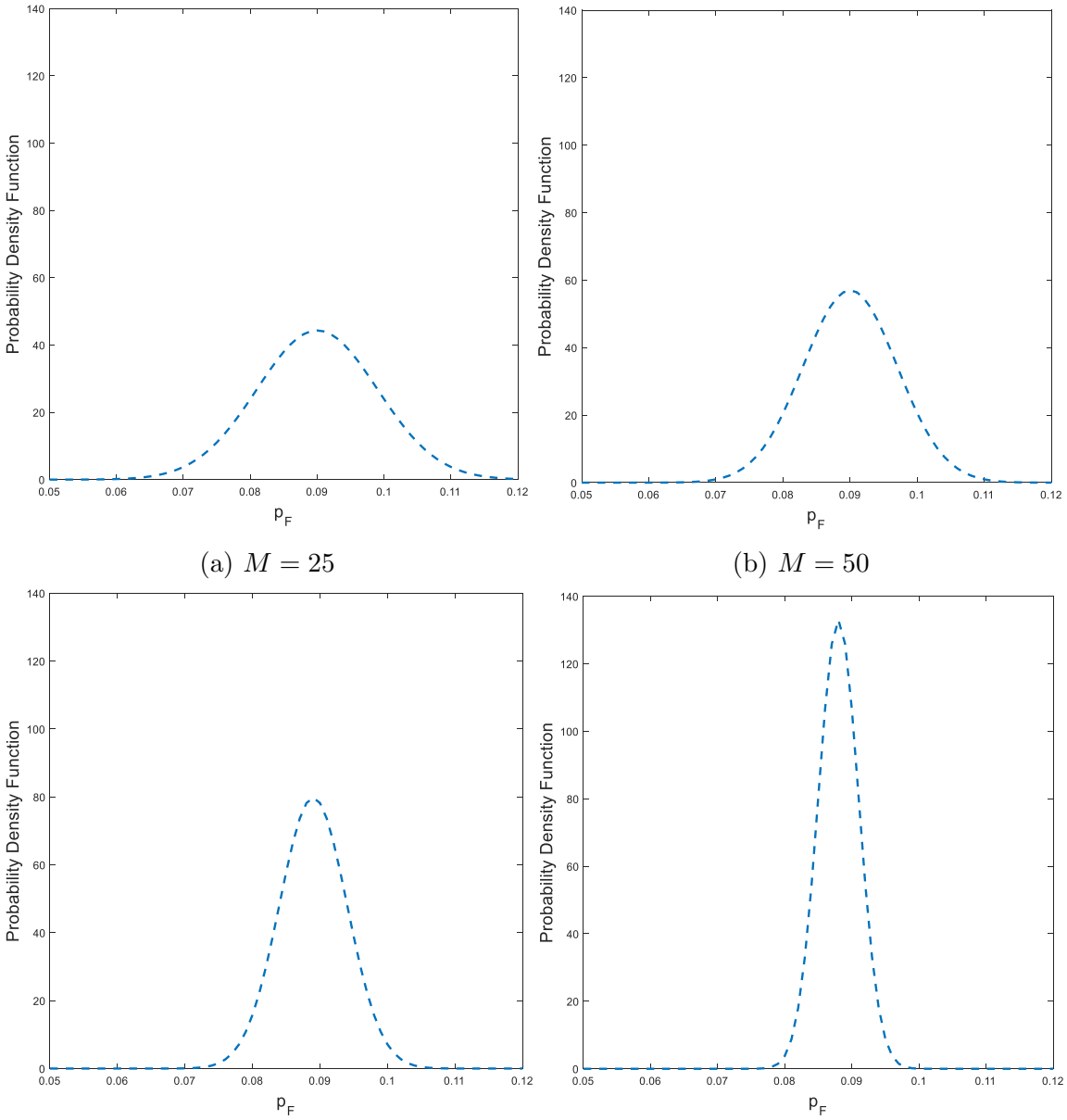


Figure 7.4: Bootstrap Confidence Intervals at Different Iterations of the Algorithm



(a) $M = 25$ (b) $M = 50$

(c) $M = 75$ (d) $M = 100$

Figure 7.5: Estimated Failure Probability Uncertainty Distribution for Different Number of Identical ANNs

7.3 Sensitivity Analysis of the SIXEP Using the Proposed Approaches

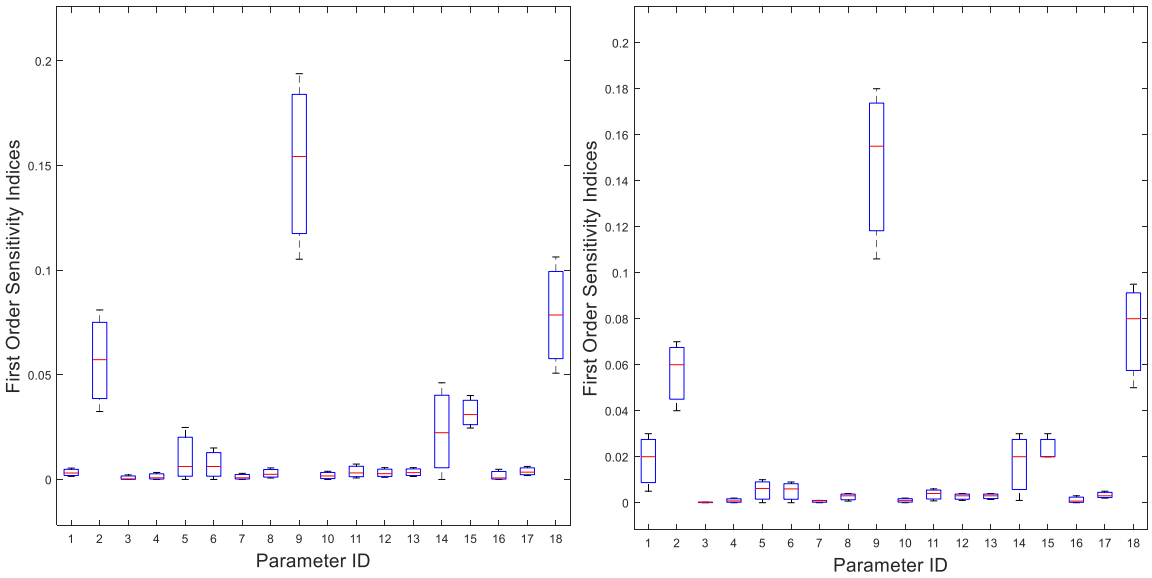
Here, the variance based approach to sensitivity analysis is used to identify the contributions of the model inputs that affects the variability in the quantities of interest.

7.3.1 Analysis

Once again, the ANNs constructed in Section 7.2 are being adopted for the sensitivity analysis. Again, the application settings are kept the same as in the numerical examples in Chapter 4 and 5, with both algorithms converging after $B = 350$ and $M = 105$ iterations respectively, adopting $N = 10^5$ MC samples in both algorithms.

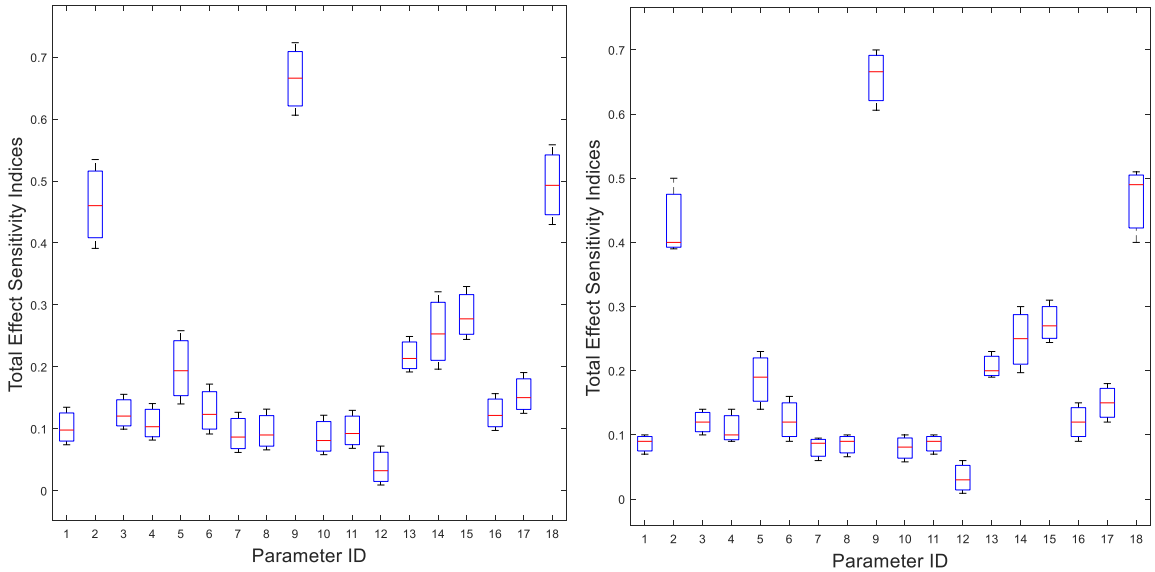
7.3.2 Results

Figure 7.6-7.9 summarizes the results obtained from both algorithms. As expected, the width of the interval Sobol' indices obtained from the approach in Chapter 4 are larger the intervals obtained from the approach in Chapter 5. The reason for the slightly wide interval obtained form the approach in Chapter 4 is as a result of the small sample size used for training, as they do not entirely capture the underlying input-output relationship of the SIXEP model, thus the margin of uncertainty is larger.



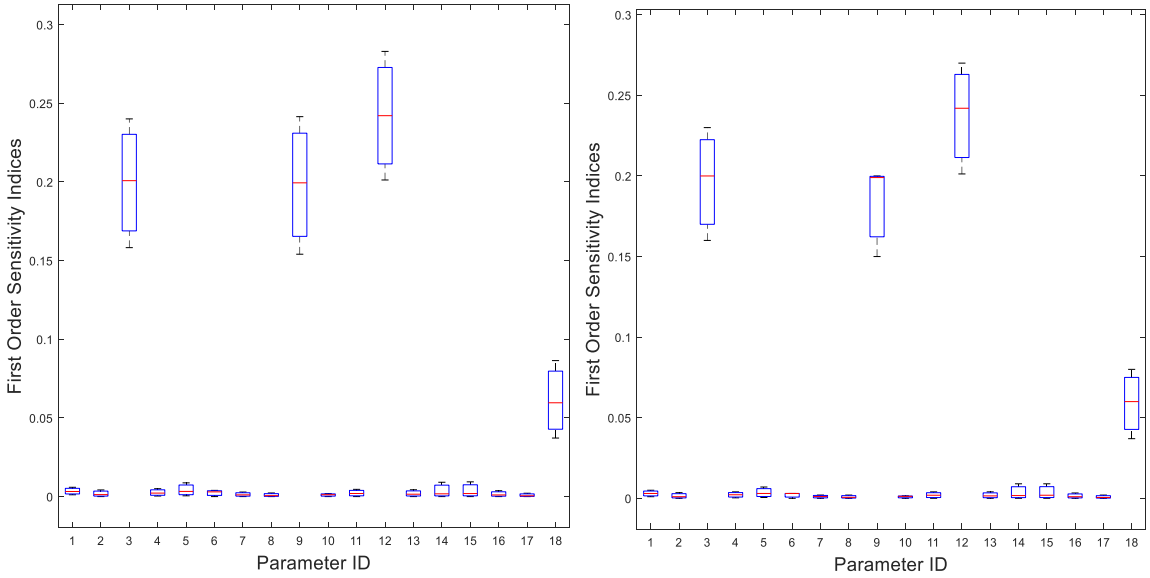
(a) Approach in Chapter 4 (b) Approach in Chapter 5

Figure 7.6: Confidence Intervals Caturing the Model Uncertainties for First Order Sensitivity Indices Estimate of ^{137}Cs



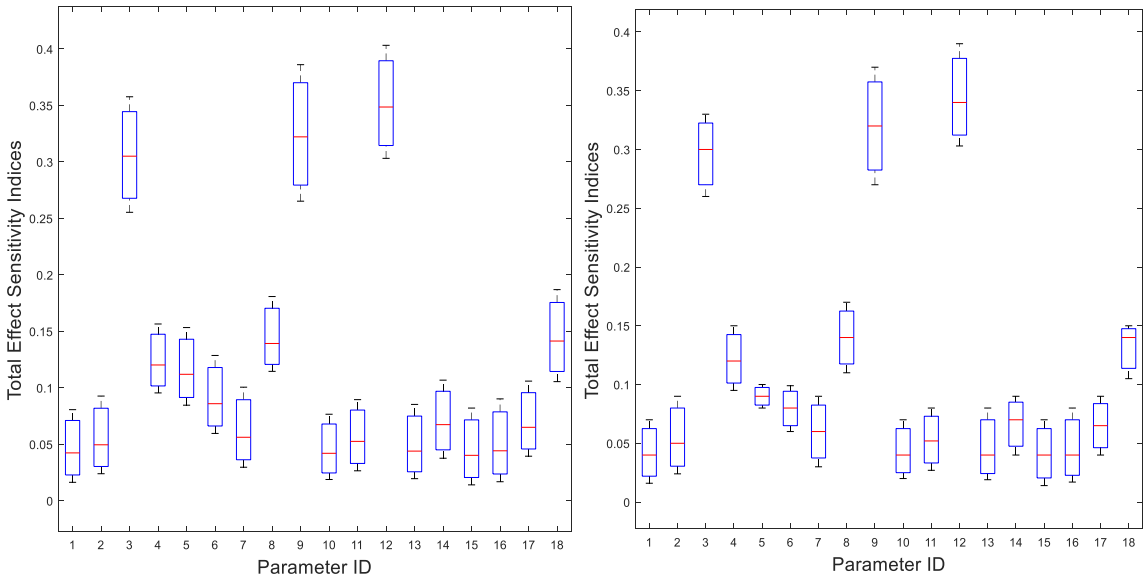
(a) Approach in Chapter 4 (b) Approach in Chapter 5

Figure 7.7: Confidence Intervals Caturing the Model Uncertainties for Total Effect Sensitivity Indices Estimate of ^{137}Cs



(a) Approach in Chapter 4 (b) Approach in Chapter 5

Figure 7.8: Confidence Intervals Caturing the Model Uncertainties for First Order Sensitivity Indices Estimate of ^{90}Sr



(a) Approach in Chapter 4 (b) Approach in Chapter 5

Figure 7.9: Confidence Intervals Caturing the Model Uncertainties for Total Effect Sensitivity Indices Estimate of ^{90}Sr

Additionally, Tables 7.2 and 7.3 reports the predictive capability in terms of the percentage of true values that falls inside the confidence bounds for the different ANNs modelling approaches.

Table 7.2: Percentage Accuracy of Chapter 4 Approach

Approach	Accuracy	RMSE
B=50	73%	0.095
B=100	82%	0.082
B=200	94%	0.076
B=300	98%	0.055

Table 7.3: Percentage Accuracy of Chapter 5 Approach

Approach	Accuracy	RMSE
M=10	55%	0.23
M=20	75%	0.15
M=50	85%	0.092
M=100	98%	0.055

7.4 Conclusion

Due to the computational challenges faced when using the SIXEP model for uncertainty quantification analyses, the proposed surrogate modelling approaches developed in Chapter 4 and 5 are adopted to alleviate these computational restrictions. Furthermore, in all the uncertainty quantification analyses considered, confidence bounds that quantifies the a specific uncertainty source have been derived. Although no reference solution is used to validate the results of both approaches on this case study, the quality of both surrogate model prediction are being tested based on the predictive performance of samples not in the training set.

Chapter 8

Fault Diagnosis of a Nuclear Power Plant Using Robust Artificial Neural Networks

ANNs are among the most powerful algorithms for regression modelling. Whereas, feed-forward artificial neural networks (FF-ANNs) (note that in this thesis ANNs and FF-ANNs are used interchangeably) can model static input-output mappings but do not have the capability of reproducing the behaviour of dynamic systems, Recurrent Neural Networks (RNNs) are attracting significant attention, because of their potentials in temporal processing. In this chapter, FF-ANNs, and RNNs are compared in the task of predicting the break size of the coolant transporting pipe in a nuclear reactor. In particular, the approaches developed in subsequent chapters are adopted to the ANN models for an adequate quantification of model uncertainties.

8.1 Background to Problem

Nuclear power plants are highly complex systems that are monitored and operated by humans. When these systems are faced with an unplanned transient, such as a plant accident scenario, equipment failure or an external disturbance to the system, the operator has to carry out diagnostic and corrective actions based on the process instrument readings. Depending on the severity of an accident, instrumental reading might not give clear indication of the developing situation. Therefore, it is necessary to develop a system that will assist the operator in order to identify such transients at the earliest stages of their developments. The objective of the plant diagnostic system in any potentially unsafe scenario is to give the plant operators appropriate inputs to formulate, confirm, initiate and perform the corrective actions.

8.1.1 Case Study: The Indian Pressurized Heavy Water Reactor (PHWR)

The Nuclear reactor considered for analysis in this chapter is the Indian PHWR. The primary heat transport (PHT) system of $220MW$ Indian PHWRs is shown in Figure 8.1. The Indian PHWR of the current design has a double containment with a vapour suppression pool. The inner primary containment (PC) is surrounded by outer secondary containment (SC) and equipped with containment engineered safety features (ESFs) like charcoal and high efficiency particulate air (HEPA) filters with containment coolers and blowers.

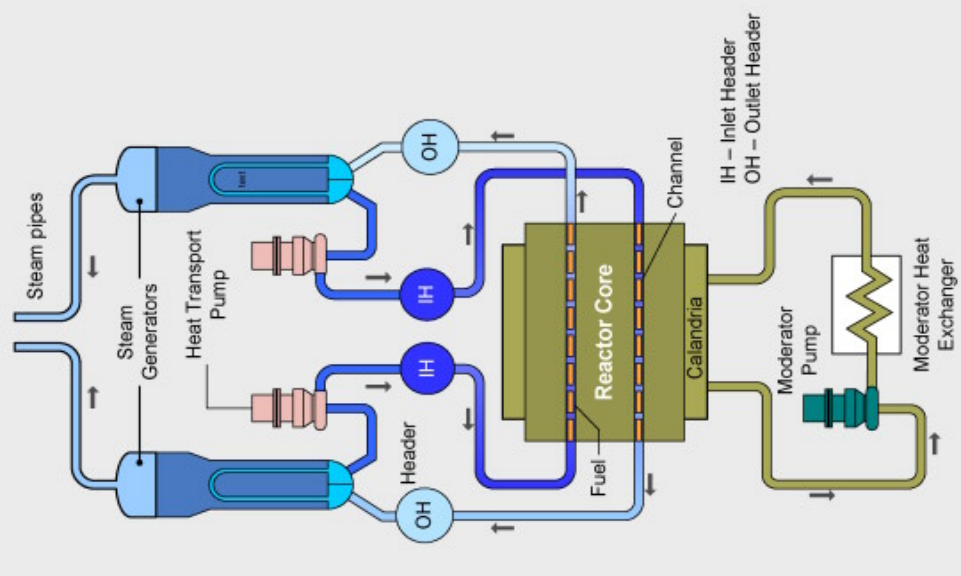


Figure 8.1: PHT system of Indian PHWR

The containment design consideration includes energy deposition from loss of coolant accident (LOCA) and main stream break line (MSLB). Most importantly, one of the major objectives of the containment is to limit the release of radioactivity at the ground level as well as through the stack within the permissible limits both during normal operation and under accident conditions. In the context of design requirements for limiting radioactivity release, the accident scenario considered in this chapter is a LOCA involving a double ended guillotine rupture of the reactor inlet header (RIH). The resulting flashing of high enthalpy liquid into the volume $V1$ will lead to its pressurisation. The pressure differential between volumes $V1$ and $V2$ causes the water column in the vents to reduce. Once the vents are cleared, it establishes the steamair mixture flow from $V1$ to $V2$. The steamair mixture bubbles through the pool where the steam gets condensed completely and the hot air is cooled

before passing to volume $V2$. The vapour suppression pool performs the important function of energy as well as radionuclide management. As an energy management feature it limits the peak pressure and temperature in the containment following a LOCA by completely condensing the incoming steam. By limiting the peak pressure, the driving force for leakage of fission products (FPs) to environment is reduced. Radionuclide management, which is a secondary function of the vapour suppression pool, involves effective FPs removal by dissolving, trapping, entraining or scrubbing away part of the FPs that reach the pool. Indian PHWR has two shut-down systems namely, primary shut-down system and secondary shut-down system to bring the reactor to shut-down state.

8.1.2 Aim and Objectives of Chapter

The aim of the study in this chapter is to develop a robust predictive model for making predictions of double ended guillotine rupture caused by LOCA. Furthermore, as predictive model developed will be affected by uncertainties described in Chapter 3 of this thesis, the approaches developed in the previous chapters of this thesis will be adopted to quantify the model uncertainties in terms of confidence intervals.

8.1.3 Data Set for Training Predictive Model

To train the predictive model a large database of these parameters has been generated based on the parameters listed in Table 8.1. The data was generated based on the analysis of 32 break scenarios of LOCA of different sizes in inlet and outlet reactor headers with and without emergency core cooling system (ECCS). Various break sizes ranging from 20% to 200% have been considered for each case mentioned above. In each case, it has been assumed that the reactor has tripped on the first trip signal. The time-dependent transient data pertaining to the reactor core and the primary heat transport has been generated using RELAP5 [75]. The sampling interval for different transient times varies, with more dense sampling at the early stages of transient. Figure 8.2 shows the reading of some selected instruments at different transient times.

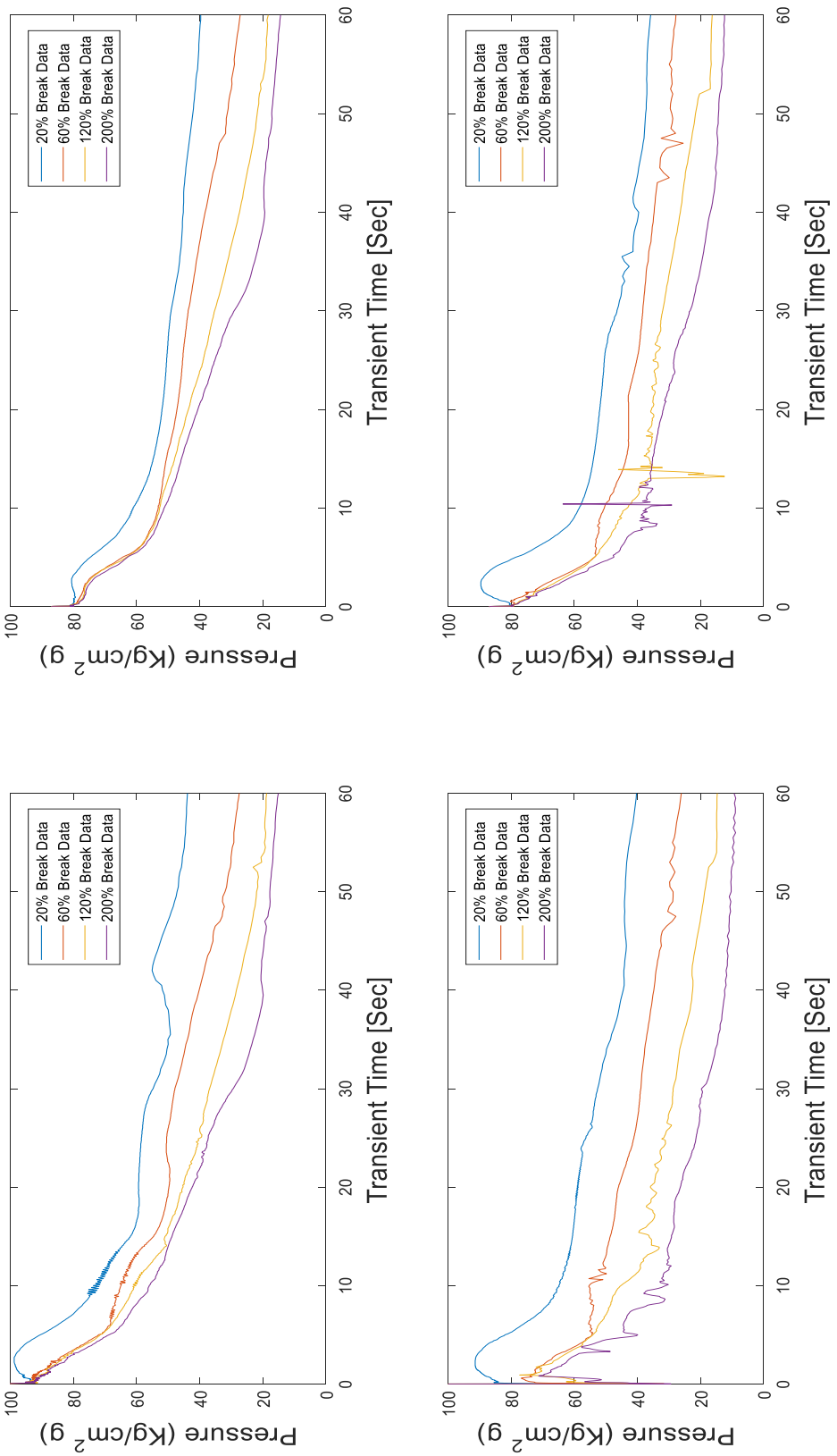


Figure 8.2: Data Samples from South Inlet Header, PHT Pressure in South Hot Header, North Inlet Header, PHT Pressure in North Hot Header Respectively.

Table 8.1: List of selected process parameters for LOCA identification

Sl. No.	Parameter description	Process range
1	South inlet header (CL) pressure CH - G	0 to 120 kg/cm^2g
2	PHT pressure in south hot header CH - D	55 to 105 kg/cm^2g
3	North inlet header (CR) pressure CH - G	0 to 120 kg/cm^2g
4	PHT pressure in north hot header CH - D	55 to 105 kg/cm^2g
5	P across header HR & CL CH - G	75 to 75 $cmWC$
6	P across header HL & CR CH - G	75 to 75 $cmWC$
7	D_2O storage tank level CH - E	0 to 4 M
8	G1 inlet temperature	0 to 320 $^\circ C$
9	SG1 outlet temperature	0 to 320 $^\circ C$
10	SG2 inlet temperature	0 to 320 $^\circ C$
11	SG2 outlet temperature	0 to 320 $^\circ C$
12	SG3 inlet temperature	0 to 320 $^\circ C$
13	SG3 outlet temperature	0 to 320 $^\circ C$
14	SG4 inlet temperature	0 to 320 $^\circ C$
15	SG4 outlet temperature	0 to 320 $^\circ C$
16	Diff temperature across SG1 CH - E	0 to 55 $^\circ C$
17	Diff temperature across SG2 CH - E	0 to 55 $^\circ C$
18	Diff temperature across SG3 CH - A/D	0 to 55 $^\circ C$
19	Diff temperature across SG4 CH - A/D	0 to 55 $^\circ C$
20	Steam flow from BO1	0 to 375,000 kg/h
21	Steam flow from BO2	0 to 375,000 kg/h
22	Steam flow from BO3	0 to 375,000 kg/h
23	Steam flow from BO4	0 to 375,000 kg/h
24	Selected channel N - 13 inlet temp.	0 to 300 $^\circ C$
25	Selected channel N - 13 outlet temp.	0 to 300 $^\circ C$
26	Selected channel H - 18 inlet temp.	0 to 300 $^\circ C$
27	Selected channel H - 18 outlet temp.	0 to 300 $^\circ C$
28	Channel flow south N - 13 CH-D	0 to 51100 kg/h
29	Channel flow south H - 18 CH - E	0 to 45200 kg/h
30	RB pump room pressure (WR)	0.1 to 5 kg/cm^2g
31	PHT pump room air temperature	0 to 100 $^\circ C$
32	D2O level in 3335 - TK1 CH - G	0 to 168 cm
33	H2O level in 3335 - TK3A & 3B CH - G	0 to 3852 $mmWC$
34	PSS linear N CH - D	0 to 150%FP
35	PSS log rate CH - D	20 to 20% s
36	PHT pump room air temperature	High (Digital)
37	PSS log rate CH - D	Trip (Digital)

8.2 Fault Diagnostic by Robust ANN

In the following sections, the approaches proposed in this thesis have been adopted to build robust ANN models are developed based on the transient data mentioned above for various postulated accident scenarios.

8.2.1 Case 1

Here, the modelling approach adopted to predict the break size is based on the approach proposed in Chapter 6. Furthermore, the effect of large and small training data sets on the robust predictions is investigated. The work flow for this section is shown in Figure 8.3.

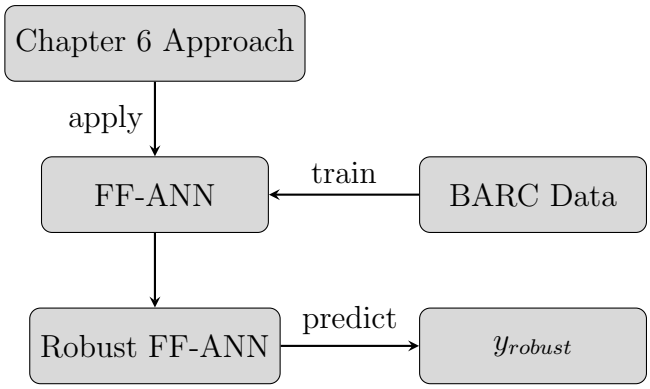


Figure 8.3: Illustration of Modelling Approach for Case 1

The detailed steps taken in this section to compute the robust break size predictions are reported below:

Step 1: Use the data provided by BARC D_{BARC} to construct the set $\mathcal{V}$ of optimal FF-ANNs using the multi-objective optimization strategy in Chapter 6.

Step 2: Construct a matrix $\mathbf{A}_{m,d}$ of size $M \times D$ containing a grid of identical optimal FF-ANNs.

Step 3: Compute the posterior probability $P(N_{m,d}|D_{BARC})$.

Step 4: Use the FF-ANNs in $\mathbf{A}_{m,d}$, to provide an estimate $\hat{Q}_{m,d}$ of the quantity Q for the given blind case input data $D_{blind\,case}$. Note that $\hat{Q}_{m,d}$ here is a vector.

Step 5: Combine the estimates $\hat{Q}_{m,d}$ to provide a robust estimate $\hat{Q}^{d}_{robust}$ of the quantity of interest $\hat{Q}$.

Step 6: Calculate the robust confidence intervals $\overline{\hat{Q}}^{d}_{robust}$ and $\underline{\hat{Q}}^{d}_{robust}$.

Step 7: Combine prediction from all the robust models using vertex method to obtain $\hat{Q}^{d}_{robust}$, $\overline{\hat{Q}}_{robust}$ and $\underline{\hat{Q}}_{robust}$.

Step 8: Check if the stopping criterion has been met. If met, proceed to the next step, otherwise, return to Step 2 and regenerate $\mathbf{A}_{m,d}$ such that the size of $\mathbf{A}_{m,d}$ is $M+1 \times D$.Note that the stopping criterion used in this present algorithm is the area a between the lower and upper confidence bound:

$$a = \sum \overline{\hat{Q}}_{robust} - \underline{\hat{Q}}_{robust} \tag{8.1}$$

and convergence is met when the tolerance $T \leq 1 \times 10^{-4}$ is achieved.

Step 9: End the algorithm.

Analysis for Effect of Large Training Data

In the multi-objective optimization problem, an initial population of 50 chromosomes have been chosen to run for 100 generations. The model architectures discovered from the optimization is given in Table 8.2.

Hidden Layers	*Activation Function*	*Neurons*	*Training Samples*
1	*Gaussian*	10	80%
1	*Gaussian*	12	80%
1	*Gaussian*	15	70%
1	*Gaussian*	18	80%
1	*Gaussian*	20	80%
1	*Gaussian*	22	70%
2	*Gaussian*	{19, 26}	80%

Table 8.2: Artificial Neural Network Architectures Discovered by Genetic Algorithm for Large Data Set.

Thereafter, the adaptive algorithm in Chapter 6 have been adopted to the FF-ANN architectures in Table 8.2 to compute intervals of various break size predictions. In Figure 8.4, a visual representation of the performance of the ANN is shown at different iterations of the algorithm.

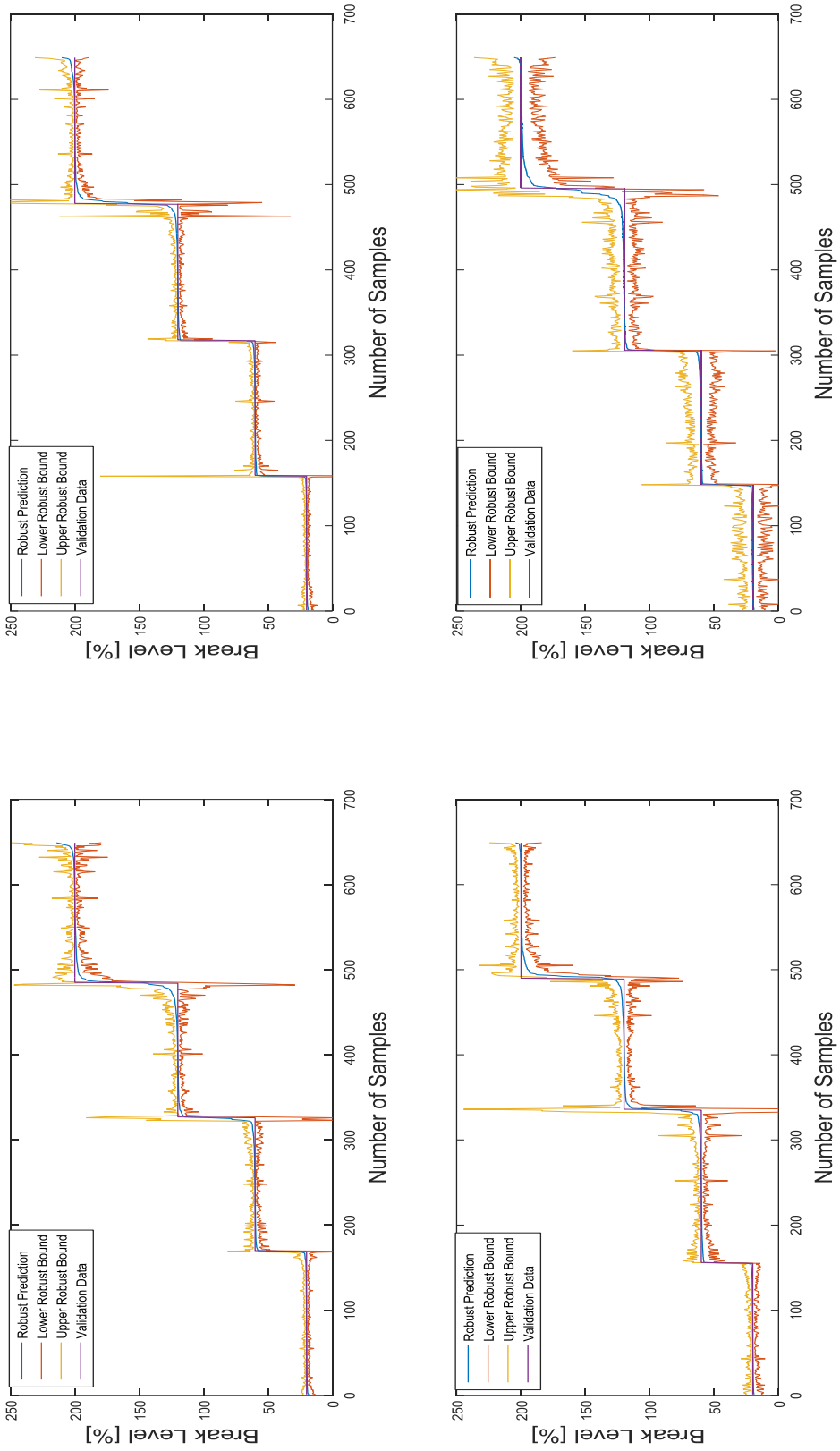


Figure 8.4: Performance of Robust ANN at Different Iterations of the Algorithm for Large Training Set. Top Left $M = 10$, Top Right $M = 20$, Bottom Left $M = 30$, Bottom Right $M = 40$.

This visual comparison gives an indication of how well the confidence bounds computed from the algorithm captures the true prediction.

Effect of Small Training Data

Similarly, the effect of small number of training samples on the confidence bounds is investigated. Here, the same experimental settings in the previous analysis have been kept. The model architectures discovered in this analysis are given in Table 8.3.

Hidden Layers	*Activation Function*	*Neurons*	*Training Samples*
2	*Gaussian*	{10, 12}	80%
2	*Gaussian*	{13,15}	70%
2	*Gaussian*	{15,17}	80%

Table 8.3: Artificial Neural Network Architectures Discovered by Genetic Algorithm for Small Data Set.

Furthermore, the adaptive algorithm have been adopted to the FF-ANN architectures discovered. Figure 8.5 shows a visual representation of the performance of the ANN at different iterations of the algorithm.

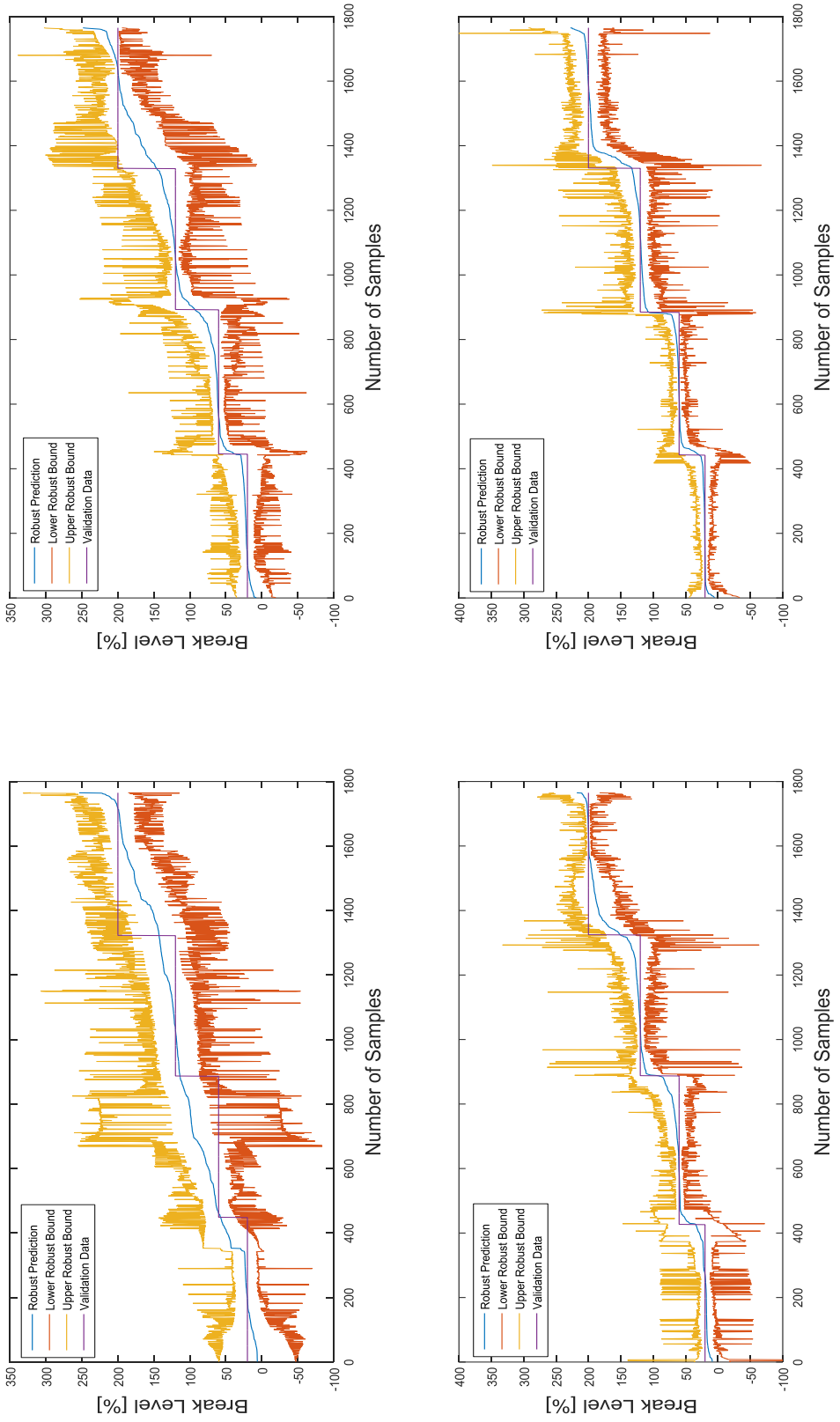


Figure 8.5: Performance of Robust ANN at Different Iterations of the Algorithm for Small Training Set. Top Left $M = 10$, Top Right $M = 20$, Bottom Left $M = 30$, Bottom Right $M = 40$.

Results: Large Training Data vs Small Training Data

In Figure 8.5 (small training data experimental result), at $M = 40$, the robust confidence bounds (although they are large) $\{\underline{y}_{robust}, \overline{y}_{robust}\}$ fully encapsulates the validation data. Also, the expected robust prediction y_{robust} closely matches the validation data. Thus, in a scenario where limited or scarce training data is provided, the approach used here can be adopted in order to compute robust confidence bounds with a high degree of accuracy. On the other hand, as more training samples are provided, the robust confidence bounds gets smaller.

8.2.2 Predicting Blind Case Data

In this section, a data set $D_{blind/case}(x)$ (i.e. data set not part of $D_{train}(x, y)$) containing only the samples input values is used to make a prediction from the robust ANN trained from the large data set. The predicted results are shown in Figure 8.6.

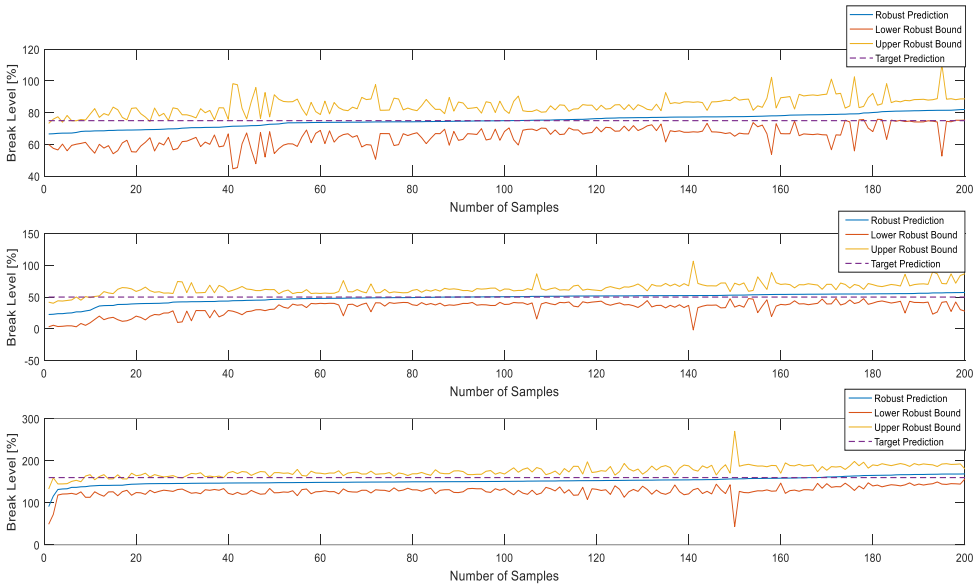


Figure 8.6: Blind Case Data Set Prediction from Robust ANN. Top Figure, Break Level Prediction for 75% Break Size. Middle Figure, Break Level Prediction for 50% Break Size. Bottom Figure, Break Level Prediction for 160% Break Size.

From the blind case prediction shown in Figure 8.6, it can be seen that the robust confidence bounds $\{\underline{y}_{robust}, \overline{y}_{robust}\}$ fully encapsulates the target prediction as the number of samples in $D_{blindcase}$ increases. Although the robust prediction y_{robust} is underestimated at a low sample size, it begins to match the target data as the sample size increases. Contrarily, for the 75% break size prediction, the robust y_{robust} value is overestimated as the sample size is increased. There could be few reasons for this

behaviour. First, due to the fact that the FF-ANN used in this analysis do not have the capability of reproducing the behaviour of dynamic systems (i.e. time variant systems), interesting features of the time series data may not be fully captured by the network. Second, although the training data set D_{train} used to train the robust ANN appears to be large, there may still be a large variability in the data set, thus, there is uncertainty in the predicted values. Thus, the possibility of exploiting deep ANN adopting the proposed techniques in this thesis will be investigated in the following sections.

8.2.3 Case 2

From literature, FF-ANNs have been found not to adequately learn important features hidden in a time series data. However, in IIR-LRNN described in Chapter 3, each layer has a recurrent connection with a tap delay associated with it. This allows the network to have an infinite dynamic response to time series data. Thus, in this section, IIR-LRNN is adopted with the approach in Chapter 6. The modelling approach adopted in this section is illustrated in Figure 8.7.

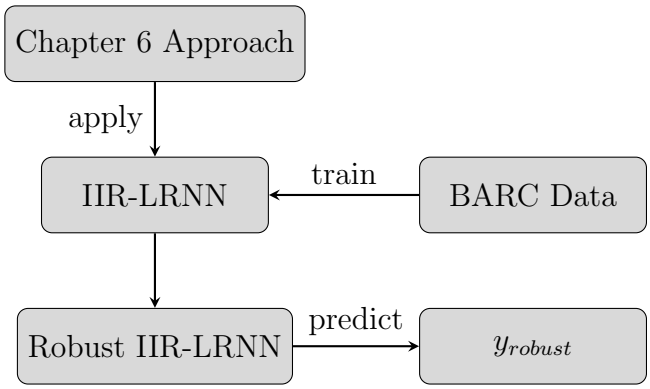


Figure 8.7: Illustration of Modelling Approach for Case 2

8.2.3.1 Adapting the IIR-LRNN to the Multi-Objective Optimization Framework

It should be noted that the multi-objective optimization framework proposed in Chapter 6 was formulated only for FF-ANN. Thus, when incorporating the IIR-LRNN into this framework, an extra modification is made. In particular, additional design variables have been added to the initial set of design variables. These additional design variables are:

- Polynomial order of moving average part in the IIR filter.
- Polynomial order of auto regressive part in the IIR filter.

Subsequently, 4 additional bits have been added to the chromosomes in Chapter 6 to make room for these additional design variables. Where information about the number of taps and delays are encoded in the first and last two additional bits respectively. This simply means that the minimum and maximum number of taps and delay falls into the interval $\{0, 4\}$. Furthermore, the adaptive algorithm in Chapter 6 is being adopted to the problem. The following are the procedures taken in this present case are given below:

Step 1: Use BARC data D_{BARC} to construct a set of optimal IIR-LRNNs using GA optimization.

Step 2: Construct a matrix $\mathbf{A}_{m,d}, m = 1, 2, ..., M, d = 1, 2, ...D$ of size $M \times D$ containing K optimal IIR-LRNNs (architecture discovered by GA) and $m \geq 2$ corresponding identical IIR-LRNNs.

Step 3: Compute the posterior probability $P(N_{m,d}|D_{BARC})$ of all the IIR-LRNNs in $\mathbf{A}_{m,d}$.

Step 4: Use the IIR-LRNNs in $\mathbf{A}_{m,d}$, to provide an estimate $\hat{Q}_{m,d}$ of the quantity Q for the given blind case input data $D_{blind\,case}$. Note that $\hat{Q}_{m,d}$ here is a vector.

Step 5: Combine the estimates $\hat{Q}_{m,d}$ to provide a robust estimate $\hat{Q}^d_{robust}$ of the quantity of interest $\hat{Q}$.

Step 6: Calculate the robust confidence intervals $\overline{\hat{Q}}^d_{robust}$ and $\underline{\hat{Q}}^d_{robust}$.

Step 7: Combine prediction from all the robust models using vertex method to obtain $\hat{Q}^d_{robust}$, $\overline{\hat{Q}}_{robust}$ and $\underline{\hat{Q}}_{robust}$.

Step 8: Check if the stopping criterion has been met. If met, proceed to the next step, otherwise, return to Step 2 and regenerate $\mathbf{A}_{m,d}$ such that the size of $\mathbf{A}_{m,d}$ is $M + 1 \times D$.

Step 9: End the algorithm.

Analysis

Similar to the previous case, the same experimental settings have been kept. After the optimization, only one IIR-LRNN architecture composed of 2 hidden layer with polynomial orders of 3 and 2 respectively in the MA and AR part have been discovered by the algorithm. Thereafter, the adaptive algorithm converged after $M = 20$ iterations.

Results

To compare the predictive capability of the robust IIR-LRNN and FF-ANN, the blind case data $D_{blind\ case}(x)$ from Section 8.2.2 is used to train both network types while adopting the proposed algorithm. The results obtained are shown in Figure 8.8. From Figure 8.8, the confidence intervals of the robust prediction have no noisy prediction when compared to the prediction made by the FF-ANN in Figure 8.6. In addition, the robust estimate from the IIR-LRNN have a lower MSE when compared to that of the FF-ANN. Furthermore, the robust confidence intervals fully encapsulates the target prediction for all samples.

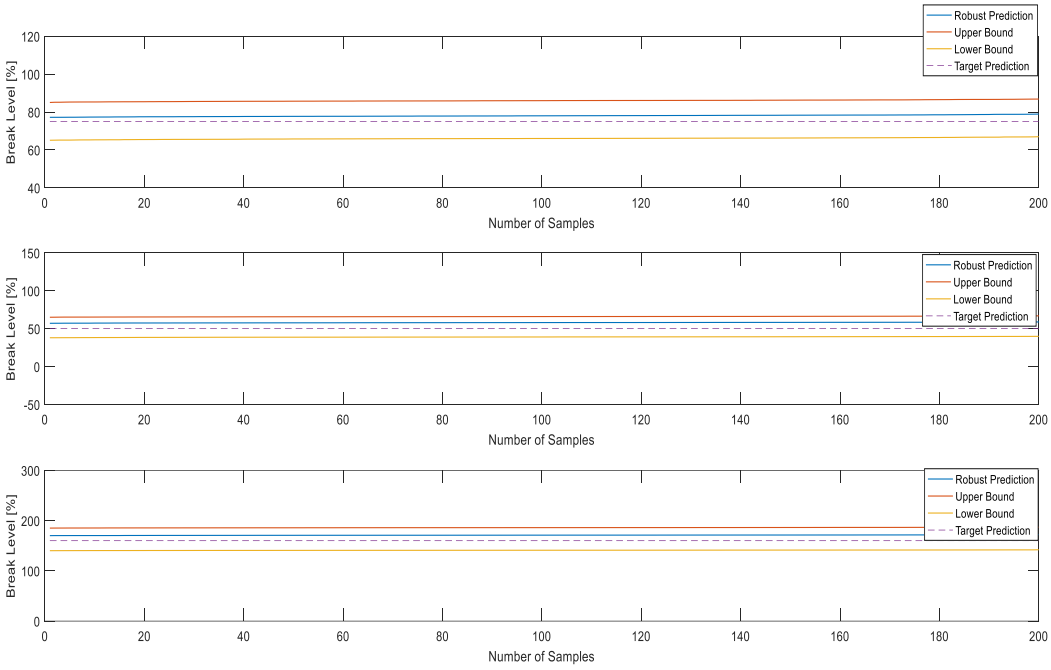


Figure 8.8: Blind Case Data Set Prediction from Robust RNN. Top Figure, Break Level Prediction for 75% Break Size. Middle Figure, Break Level Prediction for 50% Break Size. Bottom Figure, Break Level Prediction for 160% Break Size.

From these results obtained, it is clear that the robust IIR-LRNN outperformed the robust FF-ANN in terms of the predictive performance. Conversely, in this present case the width of confidence bounds obtained from the IIR-LRNN is relatively large,

having a wide margin of uncertainty. Thus, the next case considered will attempt to reduce the width of the confidence bounds (i.e. reduce model uncertainty) by combining the approach in Chapter 4 and 6.

8.2.4 Case 3

As seen in the previous section, the confidence bounds computed from the robust IIR-LRNN using the approach in Chapter 6 are wide. Thus, in this section, we attempt to reduce the confidence bounds of the robust IIR-LRNN by combining the approach in Chapter 4 and 6. The overview of the modelling approach adopted in this case is illustrated in Figure 8.9.

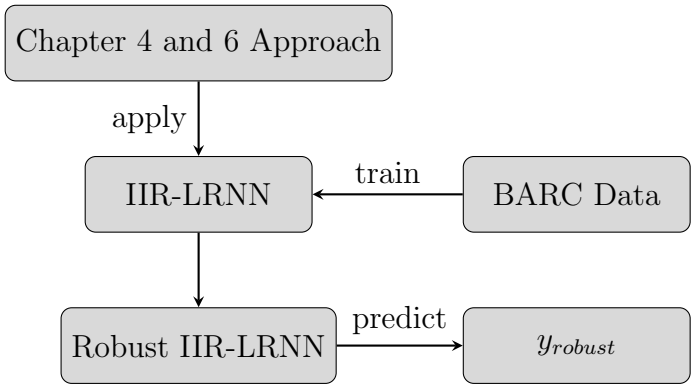


Figure 8.9: Illustration of Modelling Approach for Case 3

The proposed framework for combining this approach is given in the following section.

8.2.5 Proposed Framework for Combining Chapter 4 and 6 Approaches

First, to combine the approaches proposed in Chapter 4 and 6 for this present case, a 3-dimensional matrix $\mathbf{T}_{m,d,b}$ is introduced. Each column of $\mathbf{T}_{m,d,b}$ contains a set of unique identical IIR-LRNNs, and each page of $\mathbf{T}_{m,d,b}$ starting from the second page contains a set of corresponding bootstrapped network of the first page. The proposed algorithm for computing robust quantity of interest from the networks in $\mathbf{T}_{m,d,b}$ are reported in the following steps:

Step 1: Construct a set $\mathcal{V}$ of optimal IIR-LRNN architectures from D_{BARC}.

Step 2: Generate M identical IIR-LRNNs for each IIR-LRNN in $\mathcal{V}$.

Step 3: Construct a 3-dimensional array $\mathbf{T}_{m,d,b}$, where the first page of the 3-dimensional matrix is identical to matrix $A_{m,d}$ in Chapter 6.

Step 4: Compute the posterior probability of the IIR-LRNN in the first page.

Step 5: Provide point estimate of the quantity of interest Q in the first page of the matrix.

Step 6: Compute robust estimate $\hat{Q}^d_{robust}, \overline{\hat{Q}}^d_{robust}, \underline{\hat{Q}}^d_{robust}$ of the quantity of interest $\hat{Q}$ based on approach in Chapter 6.

Step 7: Check if the stopping criterion is met for increasing the number of M. If met, proceed to the next step, else, return to Step 2 and add another row $M = M+1$. The stopping criterion used in this present algorithm is the area a between the lower and upper confidence bound as defined in Equation **??**, and convergence is met when the tolerance $T \leq 1 \times 10^{-4}$) is achieved.

Step 8: Generate next page $B = 2$ in $\mathbf{T}_{m,d,b}$ corresponding to the bootstrap networks of the networks in the first page. Repeat step 4-6 for each subsequent page generated.

Step 9: Compute Bootstrap Bias Corrected (BBC) point estimate along the pages in $\mathbf{T}_{m,d,b}$ based on approach in Chapter 4.

Step 10: Check if stopping criterion is met for the number of bootstrap IIR-LRNNs to be constructed. If met go to next step, else, go back to step 8 and generate additional page $B = B + 1$ of bootstrap IIR-LRNNs. Note that one stopping criterion is used for all the bootstrapped models along the pages of $\mathbf{T}_{m,d,b}$, such that if any network within the array meets the criterion first, the remaining networks automatically adopts the criterion.

Step 11: Compute $\hat{Q}_{BBC,\ robust}, \underline{\hat{Q}}_{BBC,\ robust}, \overline{\hat{Q}}_{BBC,\ robust}$ such that:

$$Q_{BBC,\ robust} = \frac{1}{M} \sum_{m=1}^{M} \hat{Q}_{BBC,\ robust,\ m} \tag{8.2}$$

$\underline{\hat{Q}}_{BBC,\ robust}, \overline{\hat{Q}}_{BBC,\ robust}$ are propagated via vertex method.

Step 12: Stop the algorithm.

Subsequently, it is obvious that the training of the models and calculation of the point estimates $Q_{m,d,b}$ in $\mathbf{T}_{m,d,b}$ is computationally expensive. However, thanks to the advancements in high performance computing, such as the parallel computation support in modern devices like NVIDIA GPUs, the training of the IIR-LRNNs in $\mathbf{T}_{m,d,b}$ and the estimates $Q_{m,d,b}$ is fast.

Analysis

Using the same experimental settings and model architecture discovered in the previous case (Case 2), the algorithm converged after $M = 35$ and $B = 100$ iterations respectively.

Results

The result obtained from the above analysis is shown in Figure 8.10. From the results in Figure 8.10, it is clearly seen that the width of the confidence bounds from this present approach (Case 3) is narrower compared to that shown in Figure 8.8 (Case 2). Similarly, the expected robust estimates shown in Figure 8.10 closely matches to their respective target values compared to the results shown in Figure 8.8. Clearly, the reason for these low bias and variance in the robust prediction of the blind case is the fact that all the sources of uncertainty affecting a models prediction have been considered. Thus, it is been concluded that the proposed strategy in this present case is sufficient for an adequate quantification of model uncertainty.

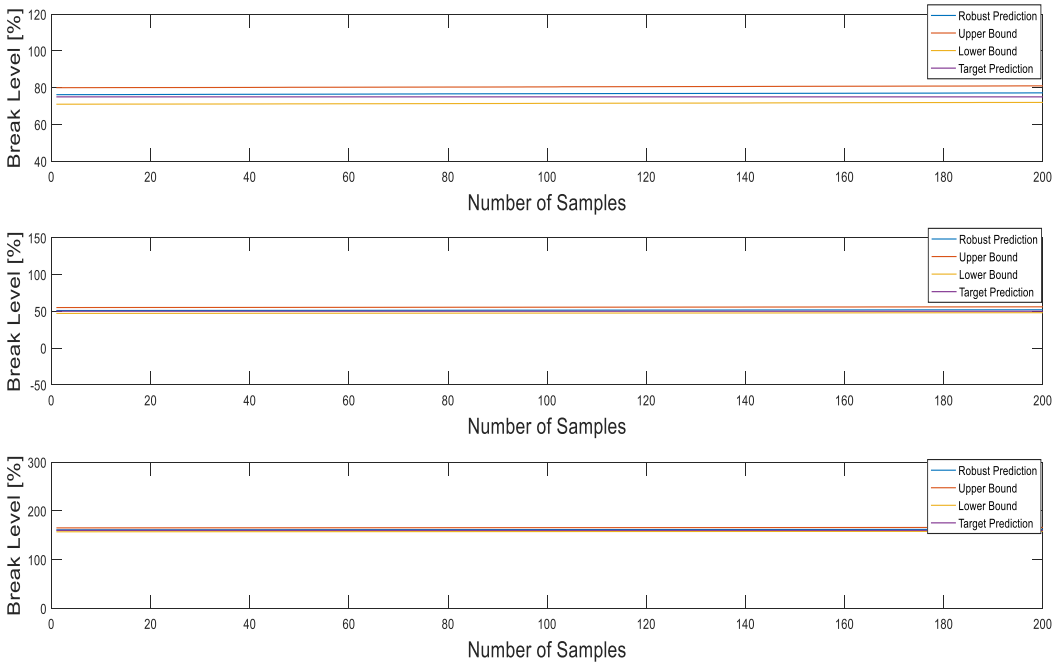


Figure 8.10: Blind Case Data Set Prediction from Robust IIR-LRNN. Top Figure, Break Level Prediction for 75% Break Size. Middle Figure, Break Level Prediction for 50% Break Size. Bottom Figure, Break Level Prediction for 160% Break Size.

8.3 Chapter Summary

In this chapter, FF-ANNs and IIR-LRNNs have been integrated into the approaches proposed in this thesis. Theses models are being compared in the task for monitoring and diagnosing a nuclear reactor based on real-time data collected from plant sensors. The models developed here carries out early detection and identifies break sizes that might affect the operation of the reactor, which might lead to core damage. In all the cases considered, IIR-LRNNs have demonstrated to outperform FF-ANNs in terms of predictive capability. In particular, the difference in the performances of the two models is much more evident in the estimation of the confidence intervals, as IIR-LRNNs always produces tighter confidence bounds compared to FF-ANNs. On the other hand, the computational time required for training the ensemble of IIR-LRNNs is much more larger than that of FF-ANNs due to the number of recurrent layers. However, with the recent advances in computational power, parallelization approaches can be adopted to reduce the computational time by a huge order of magnitude.

Chapter 9

Uncertainty Analysis of Spectral Correction Schemes in High-Energy Environments

In this chapter, surrogate models for correcting dose underestimation are proposed. In particular, the proposed models are used for correcting the readings of conventional neutron dose meters used in high energy $E_n > 10MeV$ environments. Subsequently, the approaches developed in this thesis is adopted to the proposed models in order to quantify the model uncertainties.

9.1 Problem Definition

High-energy neutrons ($E_n > 10MeV$) are relatively penetrating and usually give a substantial dose contribution behind thick shields due to their high fluence-to-dose conversion coefficients. The accuracy of neutron dose evaluation largely depends on the knowledge of neutron energy distributions at locations of concern. However, it is generally difficult to determine the spectrum over the entire energy range from thermal up to GeV neutrons. Depending on the desired energy range and resolution, various neutron detectors may have to be used in combination to achieve this goal. In radiation environments with high-energy neutrons, such as at high-energy accelerator facilities, determining the relative contribution to the total dose or dose rate from high-energy neutrons and low-energy neutrons is of great interest because high-energy neutrons may have significant contribution, but only resulting in small or negligible responses in conventional type neutron monitors.

9.1.1 Neutron Detectors for Measuring High-Energy Neutrons

Moderated-type neutron dose meters tend to underestimate the dose contribution of high-energy neutrons because of the opposite trends of dose conversion coefficients and detection efficiencies as the neutron energy increases. The phenomenon is well known to many health physics practitioners, especially those working at high-energy accelerator facilities. Improved detector designs, such as the so-called extended-range neutron dose meters have been discussed in various literatures[2, 76–79]. These extended-range neutron dose meters are relatively expensive and considerably heavy compared with the original designs due to the embedded heavy metal inside the detectors. In order to have proper dose estimation in high-energy neutron environments, it is necessary to correct the underestimated responses of the conventional neutron detectors.

9.1.2 State-of-the-art

In the study carried out in ref [2], the effect of high energy neutron spectrum on the accuracy of dose measurements was systematically investigated by considering 10 neutron spectra representing various neutron environments. Then, a spectral correction scheme was provided for users to correct the dose underestimation of conventional neutron dose meters used in radiation fields with high-energy neutrons (see Figure 9.1 for methodology adopted in [2]). It should be noted that the magnitude of correction scheme is spectrum dependent and described as a function of the estimated flux percentage of high-energy neutrons in the spectrum of workplace or a spectral index based on in situ measurements of two designated Bonner spheres. However, neutron spectra typically span several orders of magnitude and vary widely from place to place. Thus, a serious concern about the validity of the correction scheme originated from the fact that only 10 selected neutron spectra have been used to derive the correction scheme. In addition, the neutron detector and calibration source adopted in [2] might have certain effect on the estimation of dose responses and the corresponding correction factors. Hence, this present chapter aims to address these raised concerns.

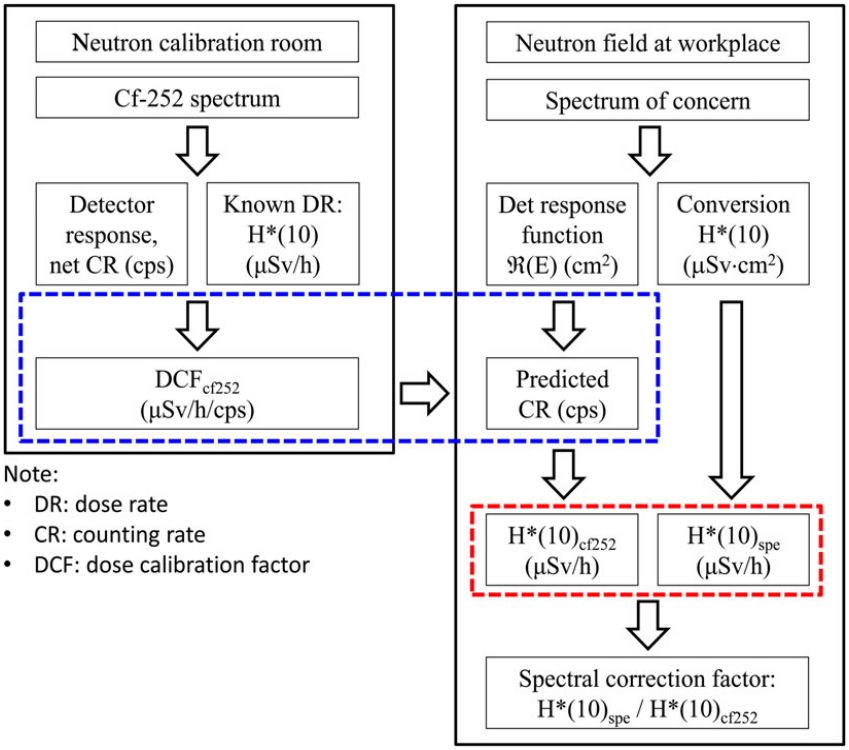


Figure 9.1: Methodology Followed in [2]

To address the aforementioned issues, the work presented in this chapter gives an improved and extended results based on a complete survey of over 200 neutron spectra collected in the IAEA Technical Reports Series No. 403 (IAEA-TRS-403)[80] and a series of sensitivity analysis to test the effect of the calibration sources on the correction scheme developed. Furthermore, the techniques developed in this thesis will be adopted to quantify the uncertainties of the correction schemes.

9.1.3 Materials and Method

9.1.3.1 Bonner Spheres and Neutron Dose Meters

Conventional neutron dose meters such as popular 9-inch rem balls and Andersson-Braun rem meters are widely used for neutron surveillance or area monitoring in workplaces. These moderated-type devices present a reasonable fit between the detection efficiency and the fluence-to-dose conversion coefficients over a wide range of neutron energies. It has been well known that the detectors based solely on moderating or absorbing materials to shape the response function suffer from no effective response to high-energy neutrons. By embedding heavy metals in neutron moderators, the effective detector response can be extended to the *GeV* range, such as

the two extended-range neutron rem meters WENDI [78] and LB6411 [76]. However, these commercial neutron dose meters, either conventional or extended-range types, were not selected in this study as enough design details such as dimensions and material compositions were not present. These details are necessary for detector modelling in numerical simulations to have an accurate prediction of the detector response function, one of the key ingredients in this study. For this reason, the Bonner sphere spectrometer have been used in this study. The Bonner sphere spectrometer is widely used in neutron spectrum determination because of several advantages including a wide energy range, isotropic angular response, reasonable detection sensitivity and excellent neutron-gamma discrimination. Intermediate-sized Bonner spheres are in principle similar to the design of most conventional neutron dose meters. More importantly, the specification of the spectrometer provides detailed information for high-fidelity response function calculations.

Figure 9.2: Bonner Sphere Spectrometer - Neutron Detector

The Bonner spheres used in this present study, consists of 5 standard polyethylene spheres of various diameters (5-inch, 6-inch, 7-inch, 8-inch, 9-inch) and two extended-range spheres. In particular, the two extended-range spheres are labelled $3P5_7$ and $4P6_8$, where the three numbers in the label indicate the diameters of the three spherical layers in inches respectively (i.e. the inner polyethylene sphere, the embedded copper (C) or lead (P) shell and the outer polyethylene sphere). The response functions (see Figure 9.3) of the Bonner spheres were calculated using the continuous-energy Monte Carlo transport code (MCNPX) [81]. The results obtained were represented in a multi-group format involving 72 equally spaced logarithmic energy bins ranging from 1 meV to 10 GeV. Depending on the availability of isotopes in libraries, the point-wise LA150 and ENDF/ B-VII.0 libraries were considered as the first and second choices of cross section data for neutron transport below certain thresholds (150

or 20 MeV). Thereafter, the default Bertini nuclear model in MCNPX was employed for simulating neutron interactions and transport. The modelling details of a sphere is essential for calculating the correct detector response function, particularly for the neutron moderator. The exact dimensions and individual polyethylene densities were used in the MCNPX calculations.

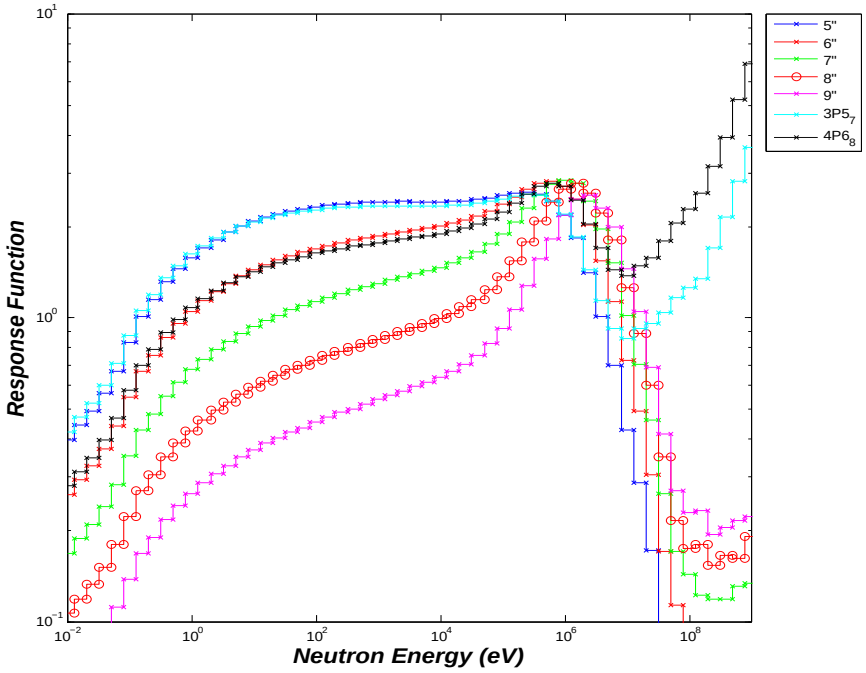


Figure 9.3: Response Function of the Various Bonner spheres used

9.1.3.2 Neutron Spectra and Dose Correction Factors

Reliable neutron dose measurement is difficult because of the wide range of neutrons and the imperfect response of most detectors. Thus, an instrument calibration is important, which requires a calibration field of similar characteristics and appropriate calibration procedure [82]. In practice, neutron dose meters used for radiation protection purpose are commonly calibrated with ^{252}Cf or other standard sources and then used in various workplaces. However, because of the limited energy range of a calibration source, calibrated dose meters are actually not recommended for use in neutron fields exhibiting characteristics that differ substantially from the calibration source. If so, one should be cautious in the detector response and a workplace-specific or spectrum-dependent correction factor may be necessary, especially for the problem of dose underestimation caused by high-energy neutrons. The approach adopted in this study for the estimation of spectral correction factors focused on three aspects of neutron dose measurement: detector calibration, response function and dose evaluation.

First, the Bonner sphere chosen as the dose meter was irradiated in a well-defined neutron field produced by a traceable standard source ^{252}Cf. The dose calibration factor of the detector expressed in unit of $Sv/h/cps$ can be determined by dividing the known dose rate at the location by the recorded net counting rate. The term dose or dose rate in this chapter refers to the operational quantity of the ambient dose equivalent, $H*(10)$. Second, by folding the neutron spectrum under consideration with the detector response function, the neutron counting rate of the detector can be estimated and further converted into the neutron dose rate. This dose rate was denoted as $H*(10)_{^{252}\mathrm{Cf}}$ because the conversion was based on the detector calibration using a ^{252}Cf neutron source. Third, the neutron dose rate at the location of interest can be evaluated by a parallel and more rigorous process, which is a direct folding of the fluence-to dose conversion factors with the spectrum. This dose rate denoted as $H*(10)_{spe}$ directly corresponds to the neutron spectrum under consideration. The ICRP-74 [83] conversion coefficients for the ambient dose equivalent were adopted for neutron energies below 180 MeV, and the high-energy extensions calculated by Pelliccioni [84] were concatenated to cover neutrons of higher energies. A comparison of the dose rates derived from the two processes leads to a spectrum-dependent correction factor for the neutron dose meter, defined as the ratio of $H*(10)_{spe}$ to $H*(10)_{^{252}\mathrm{Cf}}$. By this definition, the correction factors for neutron spectra similar to that of ^{252}Cf must be close to 1.0. For an ideal neutron dose meter, the correction factor for any given spectrum always approaches to 1.0. No spectrum-dependent correction is needed because of a perfect match between the detector response function and fluence-to-dose conversion coefficients over the entire neutron energy range. The condition obviously does not hold in reality, and in particular for high-energy neutrons. Therefore, any deviation of the calculated correction factor from the ideal value of 1.0 indicates certain spectral effect on the response of a ^{252}Cf-calibrated neutron dose meter. Through a systematic study of this effect, the relationship between the neutron field characterization and the dose response of a ^{252}Cf calibrated detector can be derived accordingly. Compared to the previous work carried out in ref [2], the value of this work lies in providing an in-depth analysis of spectral correction factors based on a much larger database. The result leads to a more rigorous and useful correction scheme than that previously provided in ref [2]. This study examined the spectral effect through a complete survey of all neutron spectra in the IAEATRS-403 report [80], rather than limited to the 10 neutron spectra that were selected subjectively. It should be noted that the IAEA report contains a large number of neutron spectra collected from various literature sources, including neutron

spectra in natural environments, neutron spectra used for instrument calibration and neutron spectra that are representative of fields in various facilities involving neutron sources or neutron-generating devices, such as nuclear power plants, medical accelerators and high-energy accelerators. Among the 243 neutron spectra collected from the IAEATRS-403 report, a total of 146 spectra contains certain portions of high-energy neutrons ($E_n > 10MeV$), ranging from a small flux percentage to 70%. Among these spectra with high-energy neutrons, 31 of them are of the most interest in the analysis because an appreciable portion of high-energy neutrons, say $\geq 10\%$, is involved in radiation fields. In addition, thousands of new spectra have been generated by a random linear combination of those spectra in the database in order to test and verify the suggested correction scheme. The correction, in essence, largely depends on the characteristics of neutron energy distribution. A systematic analysis of all these neutron spectra was performed on the basis of the detector response function and neutron field characterization.

9.1.4 Results and Discussion

9.1.4.1 Characterising the Neutron Field

If the spectrum at the location of interest is known, it is straightforward to characterize the field in terms of the flux percentage of neutrons with energies ¿10 MeV. However, if the spectrum is unknown, which is usually the case in most situations, performing radiation transport calculations or in-situ measurements are inevitable to be able to grasp some information about neutron energy distribution at the location. Alternatively, spectral indices that replace the flux percentage of high-energy neutrons have to be established. In the study [2], the pair of an extended-range sphere $4P6_8$ and a standard 6-inch sphere was selected for the purpose of constructing a spectral index, indicating the significance of high-energy neutrons in workplaces. The selection was based on an observation that the response functions of the $4P6_8$ and 6-inch spheres are nearly overlap for low-energy neutrons and deviate substantially for neutron energies ¿10 MeV (see Figure 9.3). After exploring the response functions of all Bonner sphere configurations, another pair of Bonner spheres were identified, the extended-range $3P5_7$ and standard 5-inch, which shows similar characteristics in their response functions (Figure 9.3). Therefore, the ratio between the measured counting rates of this pair of spheres could also be served as a reasonable indicator of high-energy neutrons in radiation field.

9.1.4.2 General Trends in Spectral Correction Factors

Adopting the methodology shown in Figure 9.1, a correction factor is obtained for each neutron spectrum collected from the IAEATRS-403 report. It should be noted that this correction factor is specific for a dose meter calibrated with ^{252}Cf. This spectrum-dependent correction factor, denoted as $H*(10)_{spe}/H*(10)_{^{252}\mathrm{Cf}}$, is the ratio of the ambient dose equivalent rate calculated by folding the spectrum directly with the fluence-to-dose conversion coefficients to that delivered by the detector calibrated with ^{252}Cf. Considering the standard 9-inch sphere used as a neutron dose meter, Figure 9.4 shows the distribution of dose correction factors as a function of the flux percentage of high-energy neutrons in the spectrum. Each data point in the figure represents a specific neutron spectrum collected from the IAEATRS-403 report. For those spectra without high-energy neutrons, the dose correction factors are all < 1. This reflects a well-known phenomenon that the response functions of conventional neutron dose meters tend to overestimate the magnitude of fluence-to-dose conversion coefficients for neutrons in intermediate energy range. A conservative estimate of the neutron dose in workplace is acceptable for radiation protection purposes. However, as shown in Figure 9.4, the dose correction factors for the standard 9-inch sphere used in radiation fields with high-energy neutrons may range from 1 up to > 3, indicating significant dose underestimation that cannot be ignored. On the other hand, a repeated analysis was performed by replacing the standard 9-inch sphere with the extended-range $4P6_8$ sphere. The result is shown in Figure 9.5. As expected, most of the dose correction factors are close to or < 1, indicating a satisfactory performance or at least conservative responses of this extended-range dose meter when exposed in various radiation fields with high-energy neutrons. The dose correction factors in Figure 9.4 in general show a monotonically increasing trend as a function of the flux percentage of high-energy neutrons, which enables us to propose a practical correction scheme for conventional neutron dose meters used in high-energy neutron environments. Base on the method of least squares, a second-order polynomial was fitted to establish the relationship between the dose correction factor and the high-energy neutron percentage in a spectrum. Only those spectra in IAEA-TRS-403 with appreciable component of high-energy neutrons, say $\geq 10\%$ in flux percentage, were considered in the curve fitting process. The equation of the curve was forced to pass through the given point $\{0, 1\}$ in order to meet the purpose of the correction factor in the context of phenomena discussed in this paper. The resulting equation obtained is shown in Figure 9.4 and compared with [2], which was obtained based on an analysis of 10 selected neutron spectra representing various workplaces of interest. The

difference of the two fitting curves is relatively small when compared with the overall magnitude of correction. For example, the difference between two derived correction factors is only 2% for a case of neutron field having 50% high-energy neutrons, which overall corresponds to a factor of 2 correction in neutron dose estimation. Nevertheless, the new fitting curve is suggested for practical use because it was derived from an enlarged collection of neutron spectra at various workplaces. Before applying the correction scheme in Figure 9.4 to determine a dose correction factor for the 9-inch sphere responses, it is necessary to have an estimate of the flux percentage of high-energy neutrons in the neutron field. This is impractical without the information of neutron energy distribution at the location. Neutron spectrum determination in workplaces is a difficult task, time-consuming and needs expertise. In ref [2], a practical approach as an alternative to estimate the dose correction factor based on the ratio of the measured responses of two Bonner spheres ($4P6_8$ sphere versus 6-inch) was proposed. Comparing the characteristics of their response functions, this ratio can provide an indication of the significance of high-energy neutrons in a neutron field. Figure 9.6, which is similar to Figure 9.4, presents the dose correction factors for the 9-inch sphere when used as a neutron dose meter in various neutron environments. The spectrum index in the abscissa, rather than the flux percentage of high-energy neutrons, has been replaced by the ratio between measured responses of the $4P6_8$ and 6-inch spheres. The larger the ratio between the two detectors responses, the more high-energy neutrons at the location. As expected, the dose correction factors exhibit a gradually increasing trend as a function of the new spectral index. A linear curve fitting was suggested by observing the distribution of these correction factors. If necessary, the resulting equation in Figure 9.6 can provide guidance to health physicists on the proper correction of the responses of conventional neutron dose meters. Note that a linear fitting equation was used in this case instead of a second-order polynomial proposed in [2]. As compared in Figure 9.6, the difference between the two fitting curves is within $\pm$ 10% in the whole range of the spectral indexes from 1 to 3.1, representing all the neutron spectra in the IAEA-TRS-403 report.

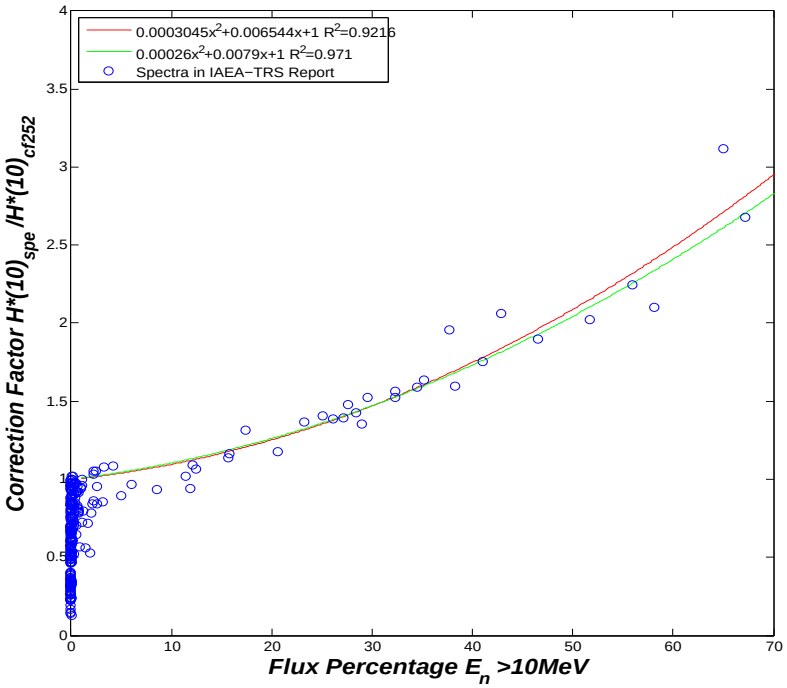


Figure 9.4: Approximation of the spectrum-dependent dose correction factors for the 9-inch Bonner sphere (calibrated with ^{252}Cf) using the flux percentage of high-energy neutrons in the spectrum and a comparison with our previous result.

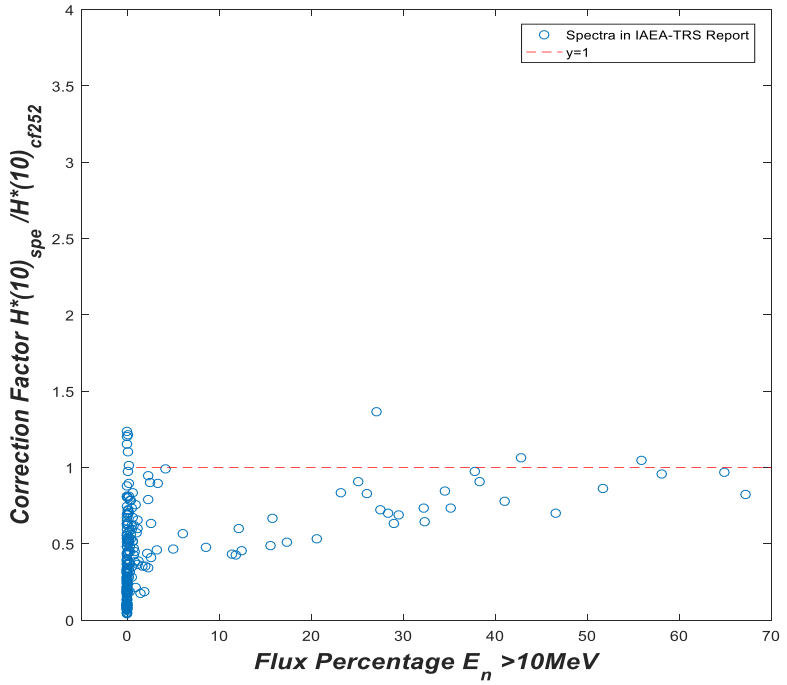


Figure 9.5: Spectrum-dependent dose correction factors for the $4P6_8$ Bonner sphere (calibrated with ^{252}Cf) as a function of the flux percentage of high-energy neutrons in the spectrum.

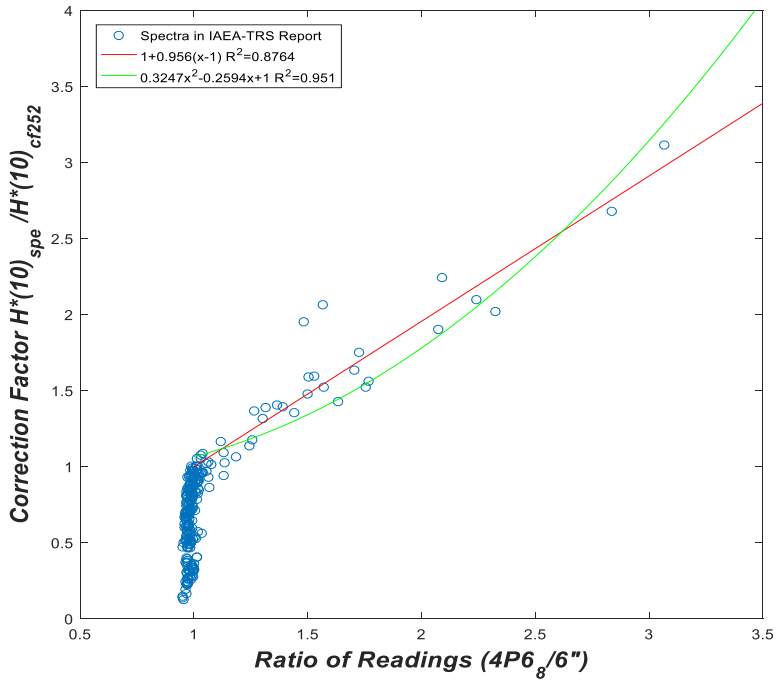


Figure 9.6: Approximation of the spectrum-dependent dose correction factors for the 9-inch Bonner sphere (calibrated with ^{252}Cf) using the ratio between the responses of two Bonner spheres ($4P6_8$ versus 6-inch) and a comparison with the result in [2]

9.1.4.3 Neutron Calibration Sources and Spectral Correction Factors

The proposed correction scheme in Figures 9.4 and 9.6 was obtained assuming that the dose meters were calibrated by a ^{252}Cf neutron source. The energy spectrum of spontaneous fission neutrons from a ^{252}Cf source can be characterized by a Maxwellian distribution and peaks at 2 MeV. ^{241}Am−Be and ^{239}Pu−Be are also commonly used neutron sources in detector calibration. Note that the two Be(,n) sources exhibit complicated spectra with multiple peaks at 3.5, 5 and 8 MeV and have higher average energies of 34 MeV. An important question arose as to what would happen to the suggested correction factors if one used different neutron sources to calibrate the dose meters. To answer this question, the procedure previously described to determine the spectral correction factors was additionally repeated twice but using ^{241}Am−Be and ^{239}Pu−Be, respectively, in place of the original calibration source ^{252}Cf. Considering the same 9 sphere as a neutron dose meter, Figure 9.7 shows a comparison of three fitting curves of dose correction factors corresponding to three different calibration sources (^{252}Cf, ^{241}Am−Be and ^{239}Pu−Be). These curves represent the suggested dose correction factors as a function of the flux percentage of high-energy neutrons in the spectrum. The data points in Figure 9.7 are spectral correction factors calculated

for a ^{252}Cf calibrated detector (same as those in Figure 9.4). The other two sets of spectral correction factors calculated for the detector calibrated by ^{241}Am−Be and ^{239}Pu−Be, respectively, are similar and omitted here, only the resulting curves are presented in the figure for clear comparison. The result in Figure 9.7 indicates that the dose correction factors are mainly a property of neutron field and not sensitive to the selection of these three commonly used calibration sources, which is a favourable outcome in practical application of these spectral correction factors.

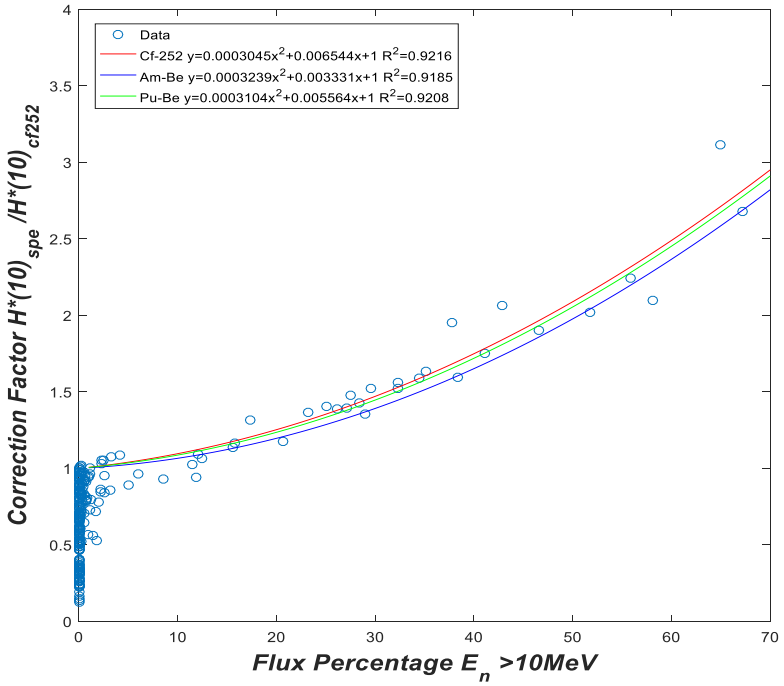


Figure 9.7: Comparison of the spectrum-dependent dose correction factors for the 9-inch Bonner sphere calibrated with ^{252}Cf, ^{241}Am−Be and ^{239}Pu−Be neutron sources.

9.1.4.4 Neutron dose meters and spectral correction factors

The spectral correction factors in Figures 9.4 and 9.6 were generated using the 9-inch Bonner sphere as a neutron dose meter. However, there are many moderated-type neutron dose meters commercially available and widely used in numerous facilities. What if one uses another dose meter with a somewhat different response function from that of the 9-inch sphere? Is the proposed correction scheme still suitable in practice? To partly address this issue, the previous procedure used to determine the spectral correction factors was repeated for neutron dose meters showing different response functions. In addition to the popular 9-inch sphere, three medium-sized Bonner spheres (6-inch, 7-inch and 8-inch) were purposely selected to represent neutron dose meters of similar type but with different response functions (see Figure 9.3).

The resulting dose correction factors were analysed and compared. As a function of the defined spectral index of high-energy neutrons in workplaces, Figure 9.8 gives a comparison of four fitting curves corresponding to four neutron dose meters (6-inch, 7-inch, 8-inch and 9-inch) under consideration. Again, the data points in the figure are spectral correction factors of the 9-inch sphere, the rest of the data points are omitted for clarity. The four fitting curves in Figure 9.8 are similar in trend with slopes varying from 0.816 to 1.010. Except for the 6-inch sphere, the spectral correction curves of the 7-inch, 8-inch and 9-inch spheres are almost consistent with each other. Comparing with the overall magnitude of the dose correction, one can conclude that the differences in these correction curves are relatively minor. For example, the difference between the resulting correction factors of two extreme spheres (6-inch versus 9-inch) is only 10% even for a neutron field with a high spectral index of 3.0, indicating a significant flux percentage, 65%, of high-energy neutrons. This observation to some extent confirmed the dose correction scheme proposed for accounting for the contribution of high-energy neutrons is dominantly a property of neutron field under consideration and only shows minor dependencies on the calibration sources and dose meters used in practical measurements. This is why sometimes it is called 'spectral correction factor' in this chapter.

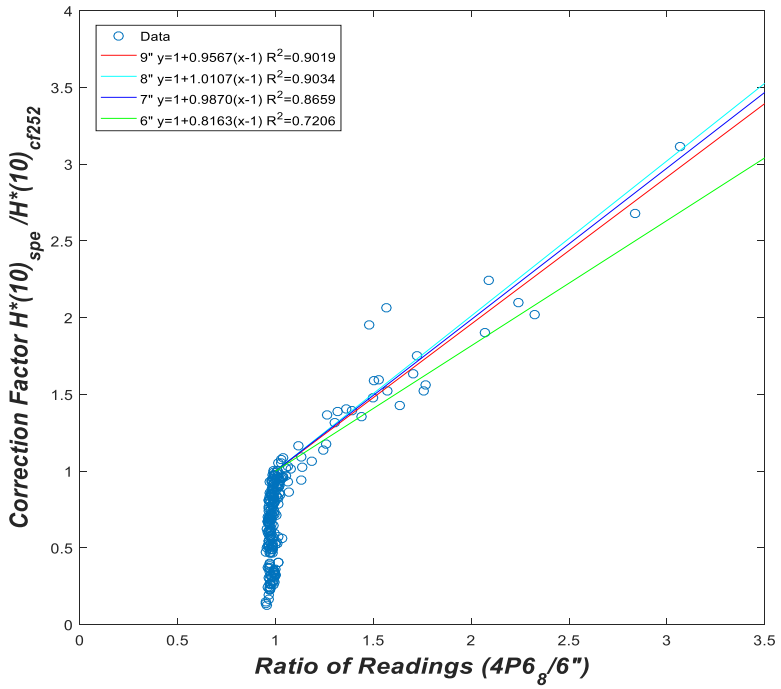


Figure 9.8: Comparison of the spectrum-dependent dose correction factors for four Bonner spheres (6-inch, 7-inch, 8-inch and 9-inch) calibrated with a ^{252}Cf neutron source.

9.1.5 Validation of the Proposed Models

Although the IAEA-TRS-403 report contains a large collection of neutron spectra available in the literature, it still does not have an exhaustive list of neutron spectra in workplaces. Hence, questions may still be asked about the robustness of the developed spectral scheme. Thus, in order to validate the proposed correction scheme, an algorithm is been proposed to construct artificial neutron spectra, then the corresponding correction factors of these spectra generated via the adopted methodology used in this chapter. By this way, all the generated new spectra could be considered at least physically meaningful and are suitable for testing the appropriateness of the proposed correction scheme. The procedures for generating the artificial high energy neutron spectra is reported in the following:

Step 1: Initialize random number generator and set $R = 1$.

Step 2: Randomly select two neutron spectra $\phi_i(E)$ and $\phi_j(E)$ with appreciable flux percentage (10%) of high-energy neutrons from the IAEA-TRS-403 collection.

Step 3: Normalize the selected spectra.

Step 4: Generate $c \sim \mathcal{U}(0, 1)$

Step 5: Superimposed both normalized spectra by applying a randomly generated weighting factor c and its additive inverse $(1 - c)$ to create a new neutron spectrum.

Step 6: Repeat steps 2-5 to generate additional spectrum.

Step 7: End algorithm.

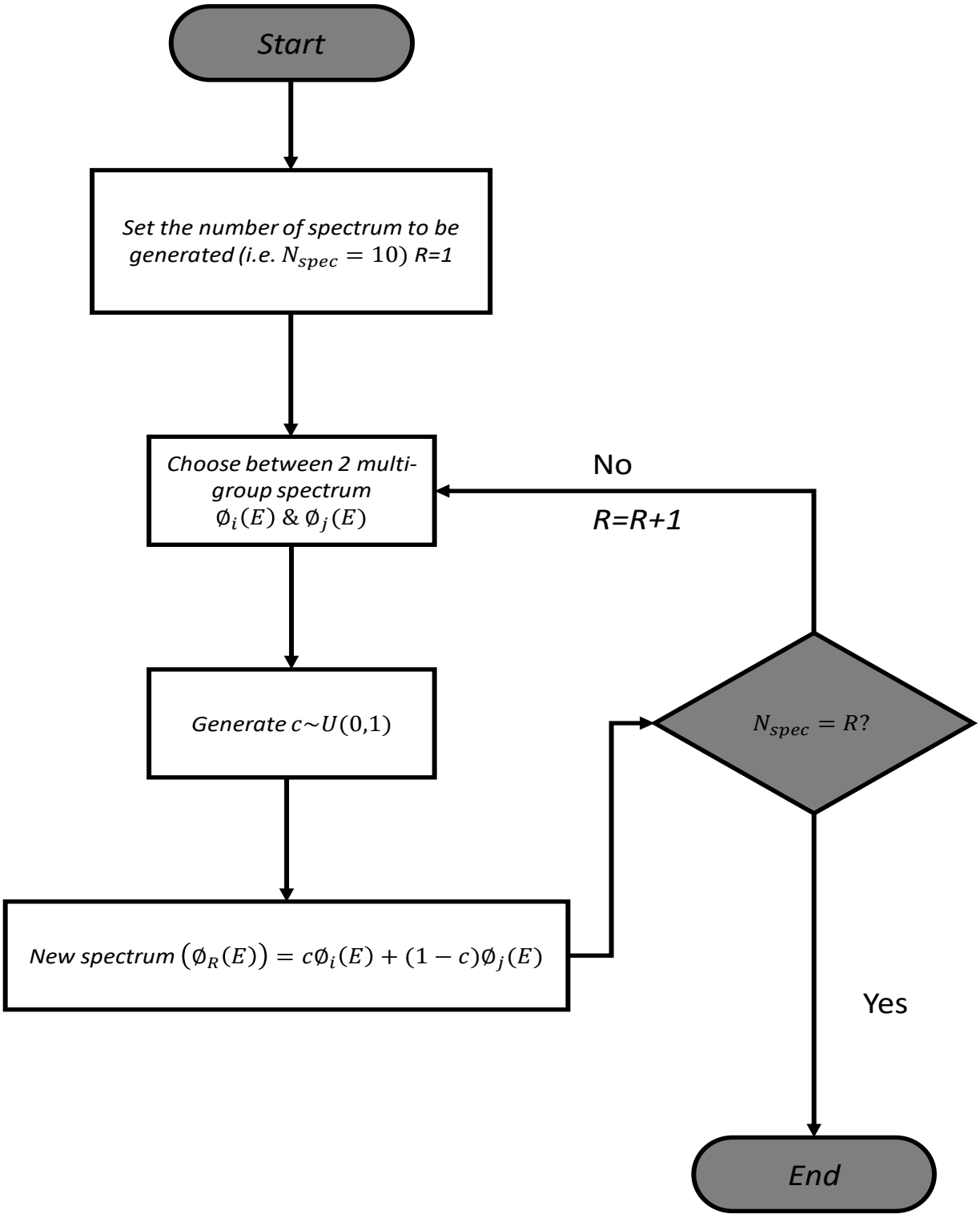


Figure 9.9: Flow Chart for Adaptive Bootstrap Algorithm

Figures 9.10 and 9.11 verify the validity of the proposed curve fitting in Figures 9.4

and 9.6, respectively. The verification process of both curves was carried out by generating randomly 1000 neutron spectra representing various workplaces.

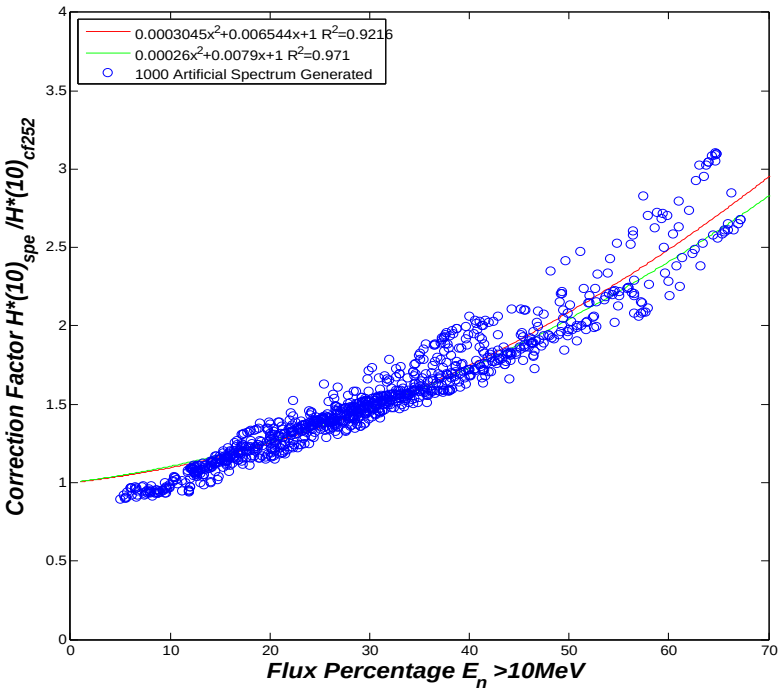


Figure 9.10: Validation of the proposed curve fitting in Figure 9.4 by considering 1000 randomly generated neutron spectra representing various workplaces.

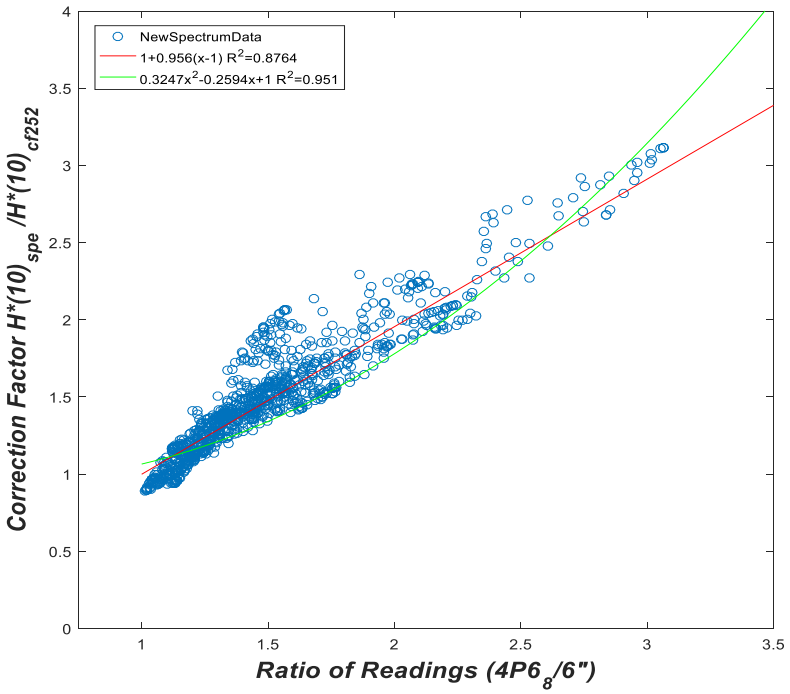


Figure 9.11: Validation of the proposed curve fitting in Figure 9.6 by considering 1000 randomly generated neutron spectra representing various workplaces.

Each spectrum can derive a spectral correction factor used to correct the under-estimated dose response of a ^{252}Cf-calibrated 9-inch Bonner sphere to high-energy neutrons. From the results shown in Figures **??** and 9.11, there is a consistent trend between the predicted curve and the spectral correction factors corresponding to those randomly generated neutron spectra. The margin of error of the proposed fitting curve in Figure 9.10 is within 10% as 90% of the randomly generated spectral correction factors fall within the error range, while the margin of error of the proposed fitting curve in Figure 9.11 is slightly larger but still within 15%. Hence, we can conclude that the proposed correction scheme matches specifications and assumptions considered acceptable for the given purpose of application.

9.2 Uncertainty Analyses of the Spectral Correction Schemes

Conversely, uncertainty affects the prediction from spectral correction schemes developed. This uncertainty arises from the variability in the data set used to develop the schemes. For example, the IAEA-TRS-403 report only contains a set of possible high energy neutron spectra from the entire population of high energy neutron spectra.

Hence, each possible set of high energy neutron spectrum can give rise to a different spectral correction scheme with different parameter. Thus, it is necessary to quantify these sources of uncertainties in order to ensure a robust reliable prediction from the develop schemes. The following section proposes an approach for quantifying these uncertainties.

9.2.1 Proposed Approach for Quantifying Uncertainty in Spectral Correction Schemes

The proposed method for quantifying the aforementioned uncertainties affecting the spectral correction model is reported in the following steps:

Step 1: Follow the methodology described in Figure 9.1 to construct the spectral correction schemes $\mu_y(x1), \mu_y(x2)$ (i.e. linear and quadratic models).

Step 2: Bootstrap each of the correction schemes from the following relationships to compute confidence bounds. The bootstrap estimate of $\hat{\mu}_y(x)$ is given by the mean provided by the ensemble of functions $\hat{\mu}_y(x^b), b = 1, 2, ...B$:

$$\hat{\mu}_y(x) = \frac{1}{B}\sum_{b=1}^{B}\hat{\mu}_y(x^b) \tag{9.1}$$

The bootstrap estimate of the standard error of $\hat{\mu}_y(x)$ is given by:

$$\hat{SE}_{boot}(\hat{\mu}_y(x)) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}[\hat{\mu}_y(x^b) - \hat{\mu}_{y,boot}(x)]^2} \tag{9.2}$$

Step 3: Compute bootstrap confidence bounds by assuming a normal distribution for $\hat{\mu}_y(x, \hat{w})$ over the space of all possible $\hat{w}$, thus, the upper bound of the bootstrap estimate are given as follows:

$$\overline{\hat{\mu}}_{boot}(x) = \hat{\mu}_{y,boot}(x) + t_{0.025[B]}\hat{SE}_{boot}(\hat{\mu}_y(x)) \tag{9.3}$$

$$\underline{\hat{\mu}}_{boot}(x) = \hat{\mu}_{y,boot}(x) - t_{0.025[B]}\hat{SE}_{boot}(\hat{\mu}_y(x)) \tag{9.4}$$

The interval $[\underline{\hat{\mu}}_{boot}(x), \overline{\hat{\mu}}_{boot}(x)]$ is a 95% confidence interval that quantifies the uncertainty in estimating the true function $\mu_y(x)$.

Step 4: Check if the stopping criterion has been met for the number of bootstrap models to be constructed. If met, proceed to the next step, otherwise, return to step 2 to generate additional bootstrap models. Note that the stopping criterion used

in this present algorithm is the area a between the lower and upper confidence bound:

$$a = \sum \overline{\hat{\mu}}_{boot}(x) - \underline{\hat{\mu}}_{boot}(x) \tag{9.5}$$

Step 5: Combine the mean and intervals predictions from the linear and quadratic models to improve the prediction from the correction schemes.

Step 6: Generate artificial spectra using the proposed algorithm in Section 9.1.5.

Step 7: Check that the R^2 value of $\hat{\mu}_y(x)$ for the artificial spectra generated is ≥ 0.9. If so, go to next step, else return to step 2 and generate $B = B + 1$ bootstrap networks.

Step 8: End the algorithm.

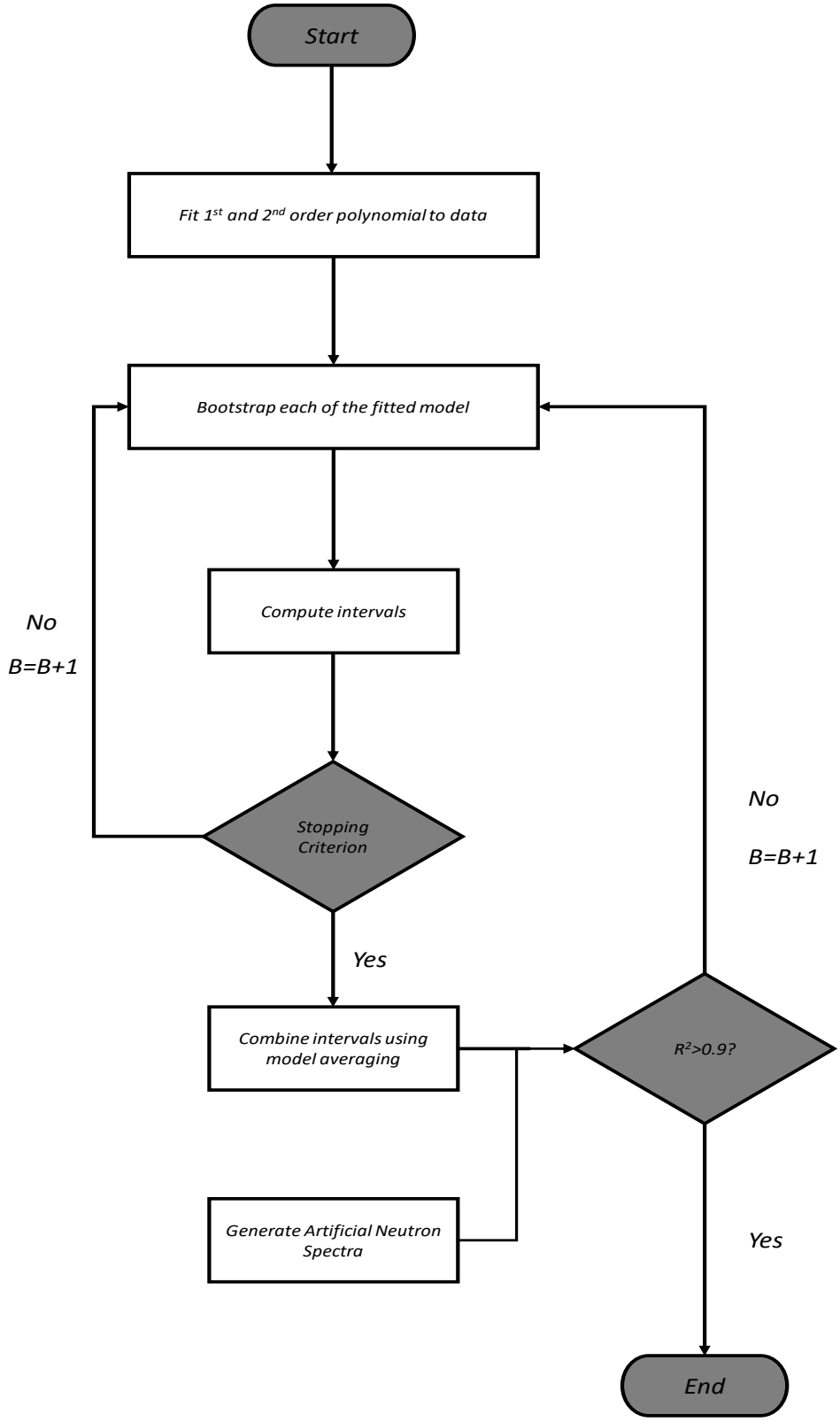
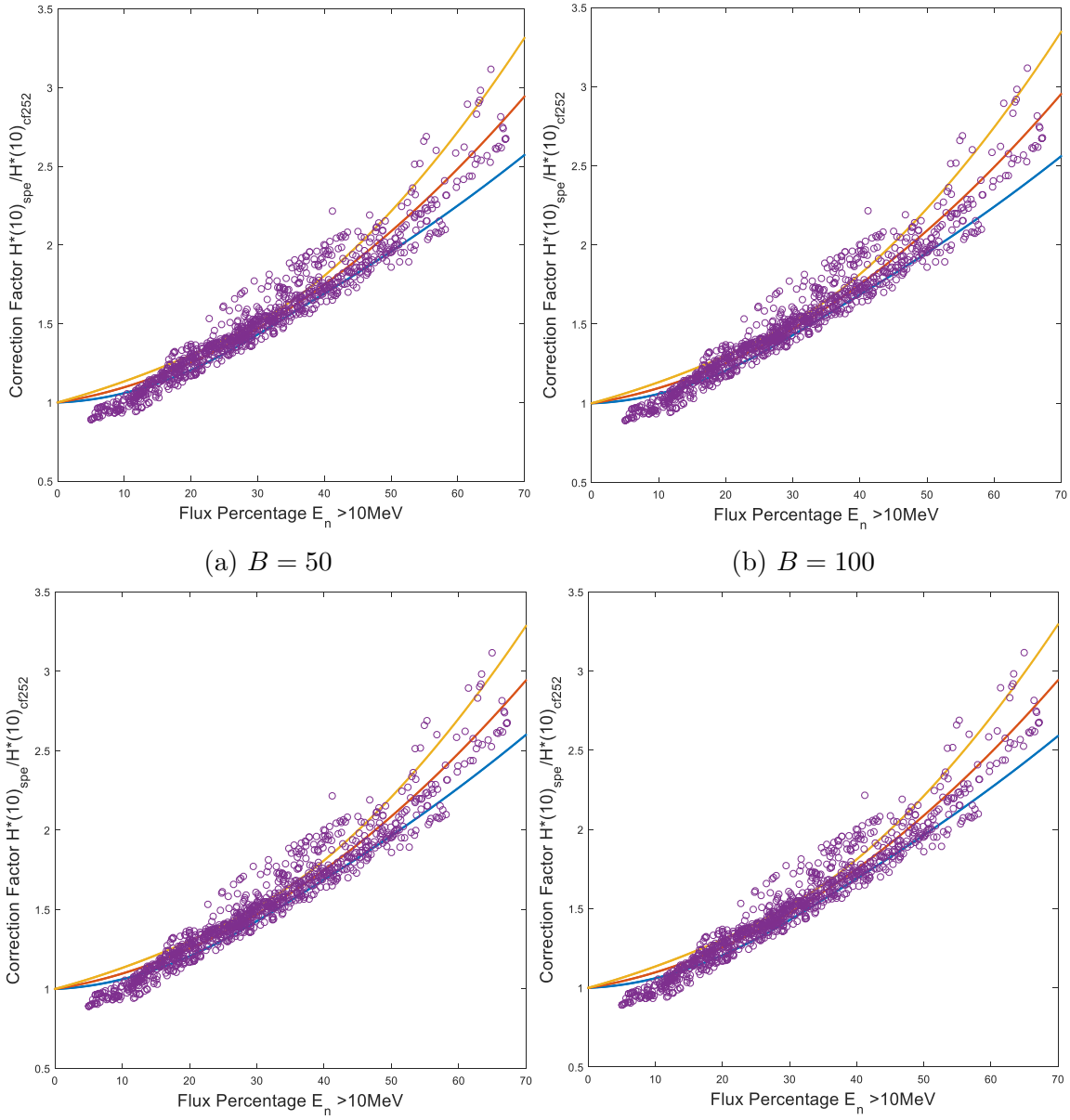


Figure 9.12: Flow Chart of Algorithm for Quantifying the Uncertainty in the Spectral Correction Schemes

Results

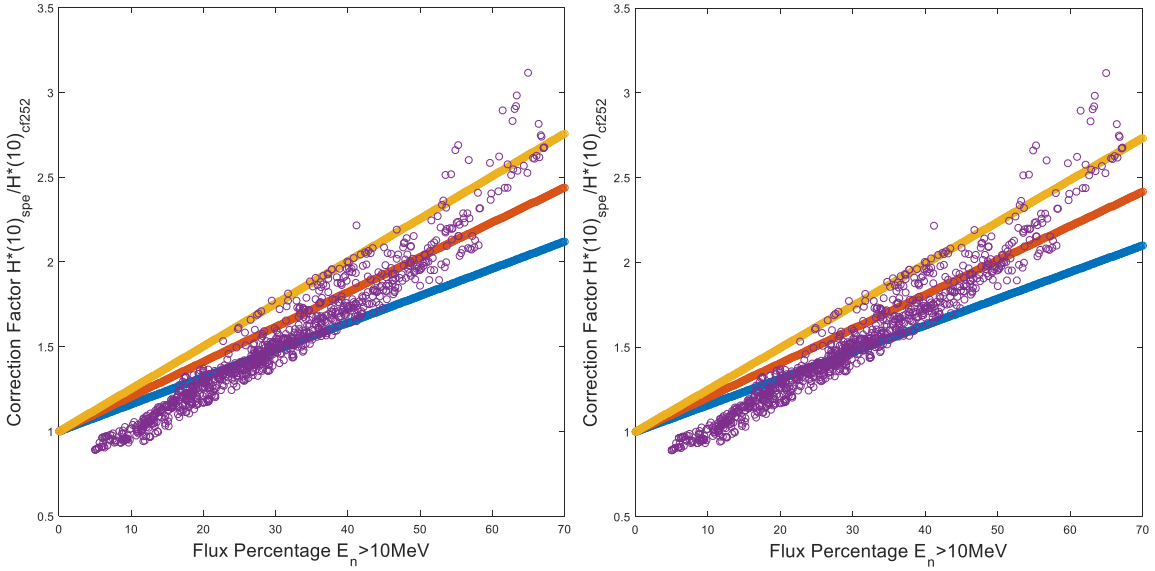
Adopting the proposed method illustrated in Figure **??**, the uncertainties affecting the spectral correction schemes have been quantified in terms of confidence intervals. Figures 9.13-9.16 shows the quantified model uncertainties for various schemes using linear and quadratic curve fitting techniques at different iteration stages of the proposed algorithm. From the figures, it can be seen that the with of the confidence intervals gets tighter (i.e. reduced uncertainty) as the number of B models are constructed. Similarly, the mean of the bootstrap models adjusts to the artificial spectra generated as B is increased. Thus, the R^2 value is increased further at higher levels of B.



(a) $B = 50$ (b) $B = 100$

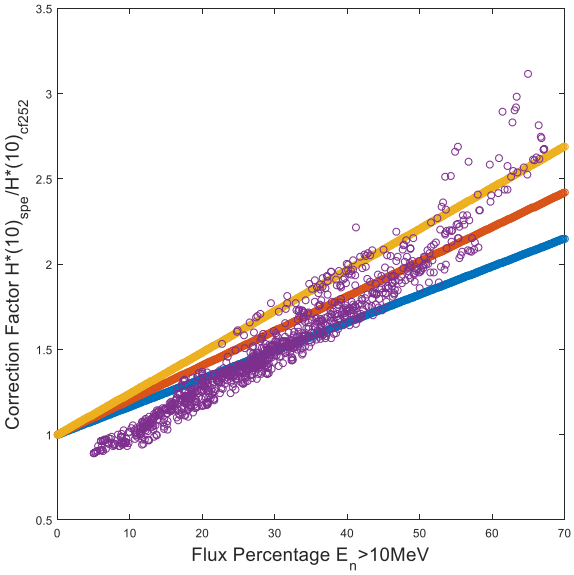
(c) $B = 200$ (d) $B = 500$

Figure 9.13: Spectral Correction Schemes (Quadratic Model) with 95% Confidence Intervals at Differetnt Iteration Levels

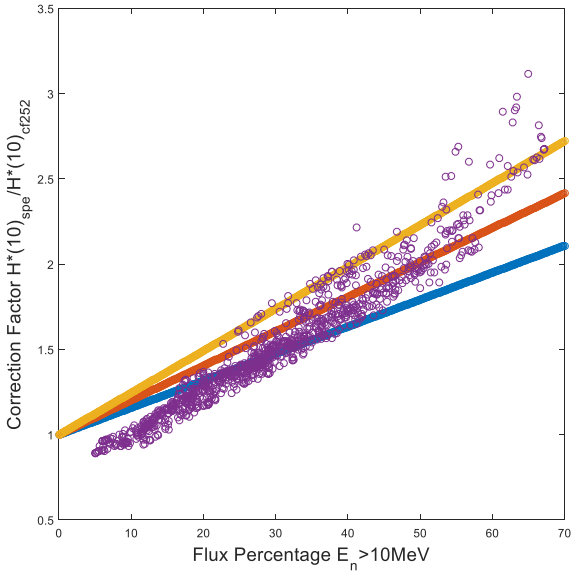


(a) $B = 50$

(b) $B = 100$

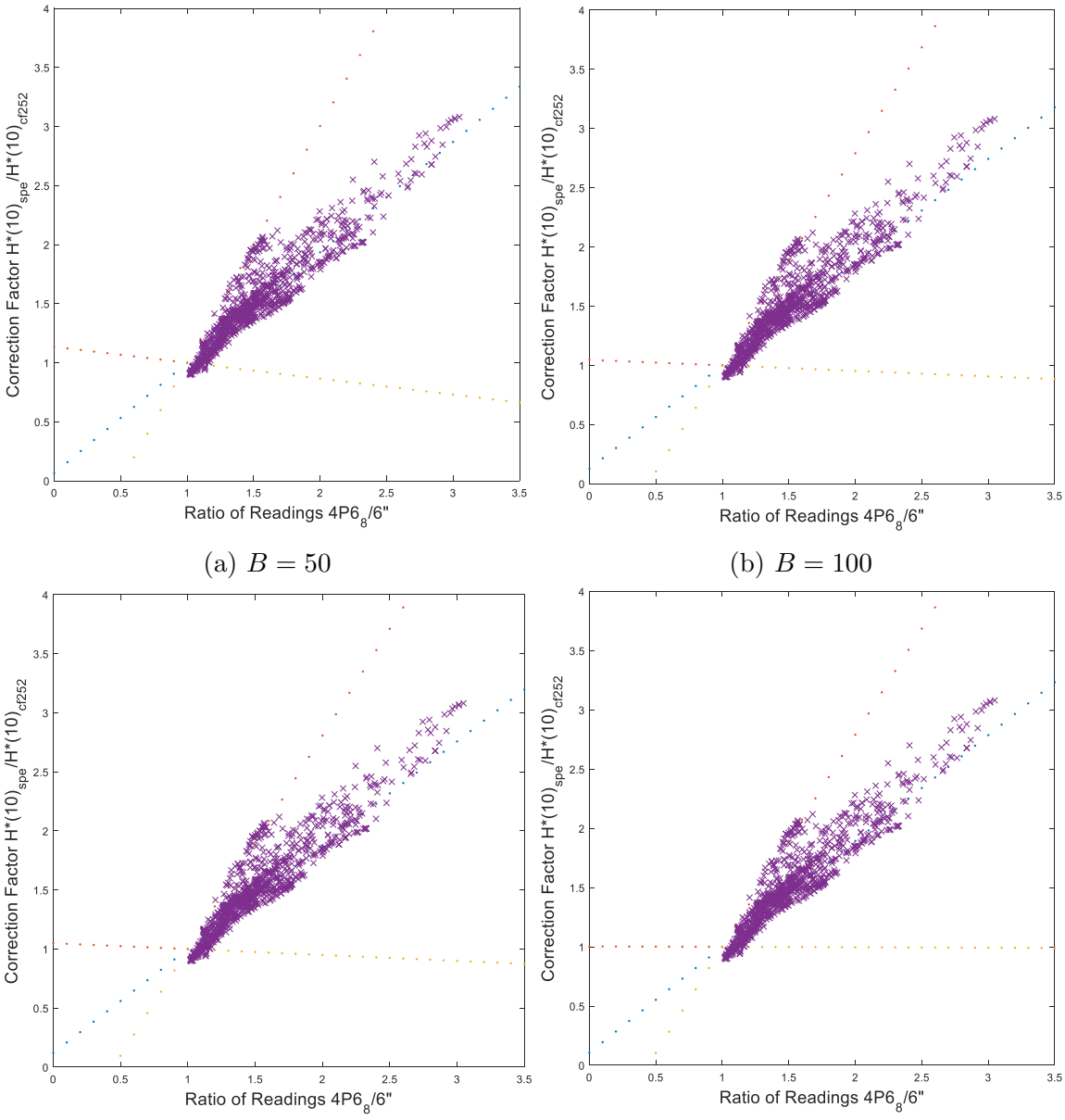


(c) $B = 200$



(d) $B = 500$

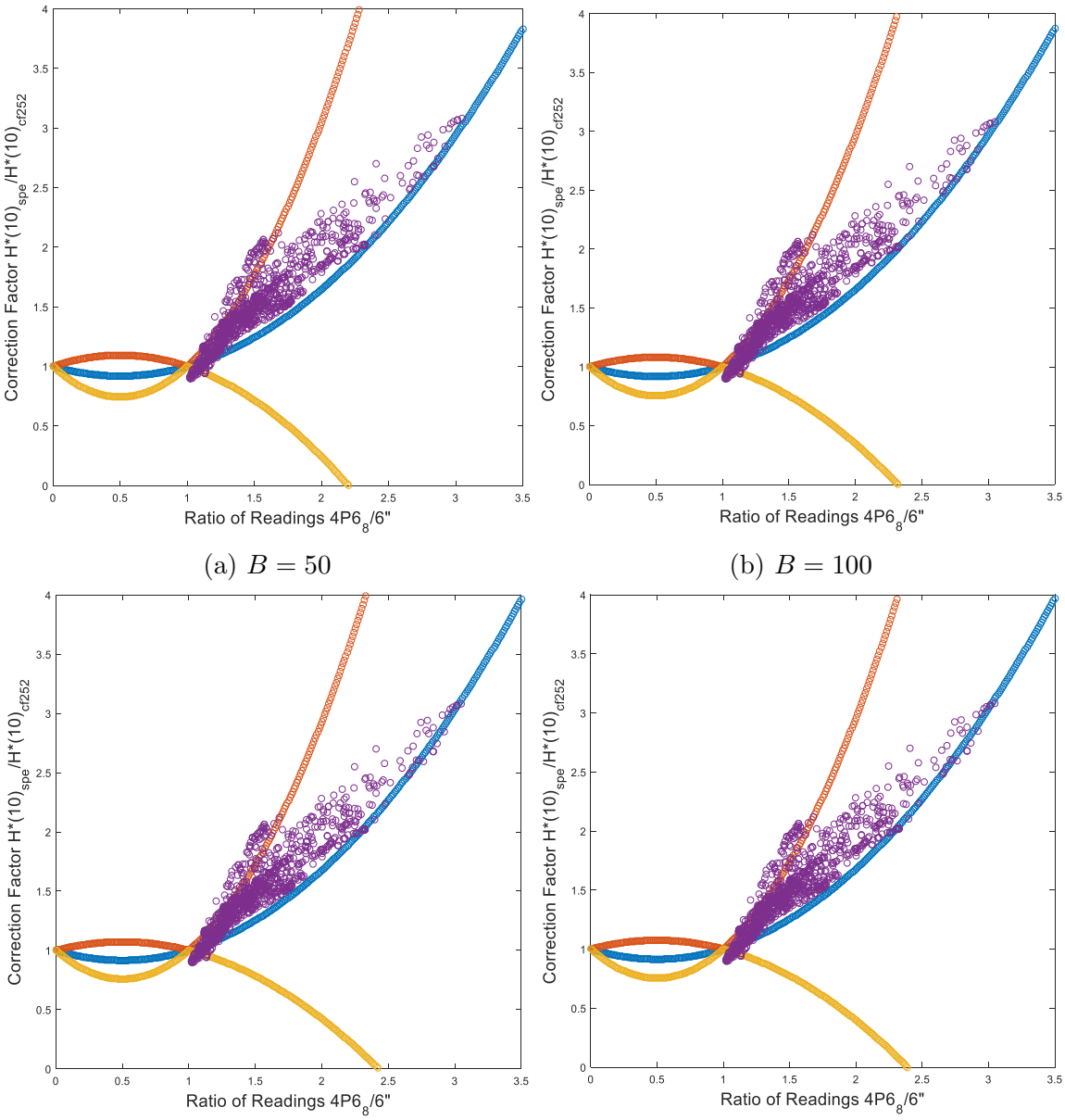
Figure 9.14: Spectral Correction Schemes (Linear Model) with 95% Confidence Intervals at Differetnt Iteration Levels



(a) $B = 50$ (b) $B = 100$

(c) $B = 200$ (d) $B = 500$

Figure 9.15: Spectral Correction Schemes (Linear Model) with 95% Confidence Intervals at Differetnt Iteration Levels



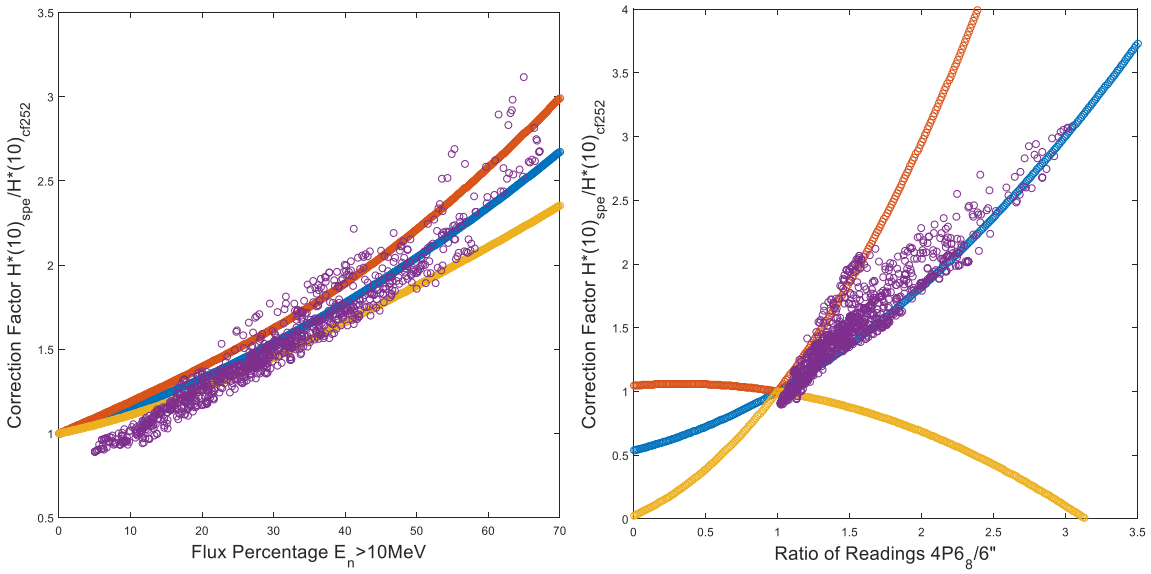
(a) $B = 50$ (b) $B = 100$

(c) $B = 200$ (d) $B = 500$

Figure 9.16: Spectral Correction Schemes (Quadratic Model) with 95% Confidence Intervals at Differetnt Iteration Levels

Model Averaging

On the other hand, the prediction intervals from the correction scheme shown in Figures 9.15 and 9.16 are relatively large compared to the Figure 9.13 and 9.14. In addition, the artificial spectra generated in Figures 9.15 and 9.16, mostly falls in the upper region of the prediction interval. Thus, both models for each respective scheme have been combined using the vertex approach to improve the overall prediction (see Figure 9.17 for results).



(a) Scheme 1 (b) Scheme 2

Figure 9.17: Model Average of the Prediction from the Linear and Quadratic Model

From the combined model prediction shown in Figure 9.17, the intervals do not show a significant reduced uncertainty, however, the mean predictions are improved in terms of R^2 values. These R^2 values are reported in Table 9.1.

Table 9.1: R^2 Values for Each Respective Correction Scheme Constructed

Scheme	Linear Model	Quadratic Model	Model Average
1	0.90	0.92	0.95
2	0.83	0.71	0.97

9.3 Chapter Summary

In this chapter, a practical scheme for correcting the dose underestimation of conventional neutron detectors in radiation environments with high-energy neutrons have been presented. The necessary correction could be significant, ranging from 1 (no correction) to more than a factor of 3 depending on the extent to which high-energy neutrons are present in radiation fields. The correction requires first a proper neutron field characterization, either in terms of the flux percentage or some spectral indexes based on in situ measurements, accounting for the significance of high-energy neutrons at the location. Fitting curves of the dose correction factors for a ^{252}Cf-calibrated 9-inch sphere dose meter were reported, as a function of the flux percentage of high-energy neutrons in the spectrum and as a function of the ratio between the measured

responses of two Bonner spheres ($4P6_8$ versus 6-inch). In addition to improved fitting results based on a large collection of neutron spectra, this study addressed two important questions associated with the applications of this correction scheme in practical situations where different calibration sources or dose meters are used. The sensitivity analysis found that different choices among three commonly used calibration sources ^{252}Cf, ^{241}Am–Be and ^{239}Pu–Be only have little effect on the values of the correction factors and the correction factors for Bonner spheres of different sizes (6-inch, 7-inch, 8-inch and 9-inch) do not change substantially, which implies that the correction factor tends to be a property of the neutron field rather than a property that strongly depends on the details of a moderated-type neutron dose meter. These observations practically facilitate the implementation and application of the suggested spectrum dependent dose correction factors in workplaces. The correction schemes developed have been fitted with 1^{st} and 2^{nd} order polynomial in order to investigate the predictive capability using both polynomials. In addition, the uncertainties in the schemes developed have been quantified using the approach proposed in this chapter. Furthermore, both models (1^{st} and 2^{nd}) for each of the respective scheme have been combined to improve the overall generalisation properties of the correction schemes.

Chapter 10

Conclusions and Recommendations

10.1 Summary

State-of-the-art engineering systems are designed to fulfil specific performance requirements despite the unavoidable uncertainty. Therefore, their respective designs should be able to deal with changing conditions driven by nature. Due to the infeasibility (i.e. huge cost, time) in testing the performance of these systems for varying levels of uncertainties, mathematical models and virtual prototypes are used in simulating the behaviour of these systems. This advance in computational development has allowed engineering practitioners to reduce the number of expensive test required to qualify a new system/product. On the other hand, a quantifiable mathematical model simulating the performance of a system is viewed to be composed of three main elements such as: 1) an input vector that represents the state variables of the system, 2) a mathematical model defining the system of interest, which is usually seen as a black-box, and finally 3) an output vector that represents the performance of the system. Under the framework of probability theory, the quantification of uncertainties requires a repeated number of model evaluations for different combinations of the probabilistic input parameters. Consequently, using the framework of the probability theory is usually expensive for complex and expensive models, thus, surrogate models (i.e. regression models) which are easy to evaluate functions are used as substitutes for these expensive models. Contrarily, the use of a surrogate model for this kind of analyses introduces additional uncertainties. Hence, it is vital to quantify these uncertainties introduced by the surrogates, in particular, when the surrogates are used by key decision makers. Attempt to quantify the uncertainties in the output of surrogate models such as ANN have been made in the past. These approaches are the delta method, based on a Taylor expansion, and the Bayesian approach, based on

Bayesian statistics to express the uncertainty of the network weights in terms of probability distributions exists. However, in the delta-method, a complex computations of derivatives and Hessian-matrix inversion is required, and the Bayesian approach require sampling from posterior weights using Markov Chain Monte Carlo (MCMC) algorithms, which are difficult to program and computationally expensive. Contrarily, the approaches proposed in this thesis are simple and relatively cheap as the require one to only train a number of models, which can be done in parallel. Subsequently, as we enter an era of high performance computing, where parallelisation on a large scale is widely accessible, the numerical approaches proposed in this thesis gains a lot of ground over the classical methods, since it can be implemented as a combination of Monte Carlo strategies and Global Optimisation methods.

10.1.1 Research Contributions

The numerical framework presented in this thesis contributes to the advancement in surrogate modelling for Uncertainty Quantification and Regression Analysis. In particular, the framework has addressed the following research questions such as:

- The quantification of surrogate model uncertainty originating from variability training data.
- The quantification of surrogate model uncertainty arising from random initialization of the weight parameters in an ANN.
- The quantification of surrogate model uncertainty arising from the model structure selected.

First, to deal with the research question about the uncertainty originating from variability in the training data set, the bootstrap technique has been employed to deal with this problem. The bootstrap method is a distribution free inference method which requires no prior knowledge about the distribution function of the underlying population. Hence, with this technique, an inference about the population where the training sample originated from can be made based on combining an ensemble of bootstrap models. Although the bootstrap technique is a well established statistical technique used in various literatures, there is no established criterion for determining the optimal number of bootstrap models to be constructed. Hence, a stopping criterion have been proposed in this thesis and adopted in the bootstrap algorithm used in this thesis. Second, to deal with the research question concerning the uncertainty arising from the random initialization of the weight parameters of the ANN due to the

training algorithm, a novel technique has been proposed to deal with this problem. The proposed technique requires training a large number of identical ANNs iteratively until convergence is met. Third, to deal with the research question concerning the uncertainty arising the model structure selected, the framework presented previously have been extended to include an optimization problem aimed at searching for optimal model structure to further reduce the bias and variance of the ANN prediction. Finally, a generalized framework has been proposed to deal with all the forms of uncertainty affecting an ANN. To demonstrate the applicability of the proposed approaches in this thesis, systems/models that are complex (i.e high dimensional), black box (i.e. only input-output relationship data is provided), and computationally expensive to run are tested.

10.1.2 Applications

The benchmark applications in some of the chapters are used to validate the proposed algorithms. They consist of mainly analytical functions or inexpensive-to-evaluate models, high fidelity computational models, and data generated from high fidelity models. Reference solutions are provided for the for the analytical or inexpensive-to-evaluate models as they are fast running models. However, in the real case studies which requires the use of a high fidelity code or data collected from specific databases, reference solutions are not provided as they are more complex and often not solvable by crude Monte Carlo simulation. Reasons are the large computational costs for a single evaluation of the computational model, as well as the large dimensionality of the input vector and scarce data. Hence, a validation set kept aside or generated artificially is used to validate the accuracy of the proposed algorithms.

Site Ion eXchange Plant (SIXEP)

The case study involving the SIXEP provides a realistic, pure black-box computational model for nuclear decommissioning purpose at Sellafield nuclear decommissioning site, UK. All aspects of the general uncertainty quantification framework are addressed, including the modelling of the input uncertainty, uncertainty propagation, structural reliability and sensitivity analysis. However, as the computational model provided is expensive, ANN have been used as a surrogate to replace the expensive model. In particular, the approaches proposed in this thesis, have been adopted to quantify the ANN uncertainty.

Fault Diagnostic of Nuclear Power Plant

This case study involve using ANN to predict the break size of a coolant transporting pipe in the event of a LOCA accident. However, a major concern regarding the use

of ANN-based monitoring and diagnostic systems in nuclear applications involved in the operation of nuclear power plants is that a quantification of the accuracy of the estimates needs to be provided. This is a crucial issue to be resolved. In this regard, the techniques developed in this thesis is used to quantify the accuracy of the estimates predicted by the ANN, while improving the robustness of the prediction.

High Energy Neutron Spectral Correction Scheme

In this case study, 1^{st} and 2^{nd} order polynomials have been used to as the surrogate model for the correction schemes proposed in this thesis. Due to the flexibility of the approaches proposed in this thesis, they have been adopted to these schemes to provide confidence intervals for each respective scheme.

Generally speaking, it has been shown by means of the application that the proposed algorithms in this thesis is capable of improving the robustness of the prediction made any type of surrogate model, while quantifying the prediction uncertainties at the same time.

Future Applications

Although the numerical approaches proposed in this thesis have been adopted to solve nuclear engineering problems, future applications lead towards the identification of opportunities for applying these developments to a large number of applications in different areas of research. This will not only facilitate the spread and adoption of the tools for general uncertainty quantification, which benefits both academia and industry, but will also increase the confidence of scientific practitioners in using surrogate models for industrial applications. Therefore, this research can seek to serve as a means to promote the use of novel numerical methods for general uncertainty quantification.

10.1.3 Future Work

The proposed algorithms offer robust tools for uncertainty quantification for a wide range of surrogate models. A number of possible extensions and applications can be envisioned, a subset of which is described in the following section:

Integration of Numerical Framework into OpenCossan A step towards the systematic use of general uncertainty quantification is been made to integrate the methods presented in this thesis in an existing open suite **OpenCossan**. The integration of the developed methods in the general framework for risk analysis and uncertainty quantification, significantly widens the spectrum of potential applications. In **OpenCossan**, a collection of predefined scripts and solution sequences is

available to facilitate connecting the new methods to real-scale problems. This makes the developments presented in this thesis of huge impact on future research.

Characterising Mixed Representation of Uncertainty In this thesis, probability theory have been used to represent the variability (aleatory uncertainty) in the parameters of the model. However, in many practical situations, the uncertainty in the parameter is of a mix of aleatory and epistemic uncertainty due to scare data. Therefore, a more generalized framework, such as Bayesian hierarchical modelling, probability box (p-box), fuzzy variable, Random set approach which are more adequate to characterize mixed aleatory and epistemic uncertainty, can be adopted and propagated through the robust surrogate models developed in this thesis.

Deep Learning with Convolution Neural Network Although recurrent neural network have shown to produce better result in terms of break size prediction, questions are still asked about how a convolution neural network may perform for this kind of case study. Thus, in the future, convolution neural network will be adopted to this problem in order to figure out the best deep neural network suited for predicting the break size more accurately.

Bibliography

[1] Scott Owens, Manon Higgins-Bos, Mark Bankhead, and Jonathan Austin. Using chemical and process modelling to design, understand and improve an efluent treatment plant. *NNL Science,*, 3,:4–13, 2015.

[2] KW Lee and RJ Sheu. Spectral correction factors for conventional neutron dosemeters used in high-energy neutron environments. *Radiation protection dosimetry*, 164(3):210–218, 2015.

[3] Marco de Angelis, Edoardo Patelli, and Michael Beer. Advanced line sampling for efficient robust reliability analysis. *Structural Safety*, 52:170–182, 2015.

[4] Andrea Saltelli, Karen Chan, E Marian Scott, et al. *Sensitivity analysis*, volume 1. Wiley New York, 2000.

[5] Edoardo Patelli. *COSSAN: A Multidisciplinary Software Suite for Uncertainty Quantification and Risk Management*, pages 1–69. Springer International Publishing, Cham, 2016. ISBN 978-3-319-11259-6.

[6] Edoardo Patelli, H Murat Panayirci, Matteo Broggi, Barbara Goller, Pierre Beaurepaire, Helmut J Pradlwarter, and Gerhart I Schuëller. General purpose software for efficient uncertainty management of large finite element models. *Finite elements in analysis and design*, 51:31–48, 2012.

[7] Alfred M Freudenthal and Masanobu Shinozuka. Structural safety under conditions of ultimate load failure and fatigue. Technical report, COLUMBIA UNIV NEW YORK, 1961.

[8] Masanobu Shinozuka. Basic analysis of structural safety. *Journal of Structural Engineering*, 109(3):721–740, 1983.

[9] Karl Breitung. Asymptotic approximations for multinormal integrals. *Journal of Engineering Mechanics*, 110(3):357–366, 1984.

[10] Edoardo Patelli, Helmut J Pradlwarter, and Gerhart I Schuëller. On multinormal integrals by importance sampling for parallel system reliability. *Structural Safety*, 33(1):1–7, 2011.

[11] GI Schueller. Efficient monte carlo simulation procedures in structural uncertainty and reliability analysis-recent advances. *Structural Engineering and Mechanics*, 32(1):1–20, 2009.

[12] RE Melchers. Importance sampling in structural systems. *Structural safety*, 6 (1):3–10, 1989.

[13] PS Koutsourelakis, HJ Pradlwarter, and GI Schuëller. Reliability of structures in high dimensions, part i: algorithms and applications. *Probabilistic Engineering Mechanics*, 19(4):409–417, 2004.

[14] Siu-Kui Au and James L Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4): 263–277, 2001.

[15] Siu-Kui Au and Edoardo Patelli. Rare event simulation in finite-infinite dimensional space. *Reliability Engineering & System Safety*, 148:67–77, 2016.

[16] Enrico Zio and Nicola Pedroni. Subset simulation and line sampling for advanced monte carlo reliability analysis. In *Proceedings of the European Safety and RELiability (ESREL) 2009 Conference*, pages 687–694, 2009.

[17] Andrea Saltelli. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications*, 145(2):280–297, 2002.

[18] Jon C Helton, Jay D Johnson, WL Oberkampf, and Cédric J Sallaberry. Sensitivity analysis in conjunction with evidence theory representations of epistemic uncertainty. *Reliability Engineering & System Safety*, 91(10):1414–1434, 2006.

[19] Chonggang Xu and George Zdzislaw Gertner. Uncertainty and sensitivity analysis for models with correlated parameters. *Reliability Engineering & System Safety*, 93(10):1563–1573, 2008.

[20] Ilya M Sobol. Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments*, 1(4):407–414, 1993.

[21] Ilya M Sobol. On quasi-monte carlo integrations. *Mathematics and Computers in Simulation*, 47(2):103–112, 1998.

[22] Jon C Helton and Freddie Joe Davis. Latin hypercube sampling and the propagation of uncertainty in analyses of complex systems. *Reliability Engineering & System Safety*, 81(1):23–69, 2003.

[23] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global sensitivity analysis: the primer*. John Wiley & Sons, 2008.

[24] RI Cukier, HB Levine, and KE Shuler. Nonlinear sensitivity analysis of multiparameter model systems. *Journal of computational physics*, 26(1):1–42, 1978.

[25] Andrea Saltelli, Stefano Tarantola, and KP-S Chan. A quantitative model-independent method for global sensitivity analysis of model output. *Technometrics*, 41(1):39–56, 1999.

[26] Stefano Tarantola, Debora Gatelli, and Thierry Alex Mara. Random balance designs for the estimation of first order global sensitivity indices. *Reliability Engineering & System Safety*, 91(6):717–727, 2006.

[27] Jean-Yves Tissot and Clémentine Prieur. Bias correction for the estimation of sensitivity indices based on random balance designs. *Reliability Engineering & System Safety*, 107:205–213, 2012.

[28] Mark J Anderson and Patrick J Whitcomb. *Design of experiments*. Wiley Online Library, 2000.

[29] Roger G Ghanem and Pol D Spanos. *Stochastic finite elements: a spectral approach*. Courier Corporation, 2003.

[30] Jerome Sacks, William J Welch, Toby J Mitchell, and Henry P Wynn. Design and analysis of computer experiments. *Statistical science*, pages 409–423, 1989.

[31] Thomas J Santner, Brian J Williams, and William I Notz. *The design and analysis of computer experiments*. Springer Science & Business Media, 2013.

[32] Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

[33] Steve R Gunn et al. Support vector machines for classification and regression. *ISIS technical report*, 14:85–86, 1998.

[34] Jooyoung Park and Irwin W Sandberg. Universal approximation using radial-basis-function networks. *Neural computation*, 3(2):246–257, 1991.

[35] Sang-Hyo Kim and Seong-Won Na. Response surface method using vector projected sampling points. *Structural safety*, 19(1):3–19, 1997.

[36] Christopher M Bishop. *Neural networks for pattern recognition*. Oxford university press, 1995.

[37] Roland Schobi, Bruno Sudret, and Joe Wiart. Polynomial-chaos-based kriging. *International Journal for Uncertainty Quantification*, 5(2), 2015.

[38] Christian Bucher and Thomas Most. A comparison of approximate response functions in structural reliability analysis. *Probabilistic Engineering Mechanics*, 23(2):154–163, 2008.

[39] Henri P Gavin and Siu Chung Yau. High-order limit state functions in the response surface method for structural reliability analysis. *Structural safety*, 30 (2):162–179, 2008.

[40] Abbie B Liel, Curt B Haselton, Gregory G Deierlein, and Jack W Baker. Incorporating modeling uncertainties in the assessment of seismic collapse risk of buildings. *Structural Safety*, 31(2):197–211, 2009.

[41] Jian Deng, Desheng Gu, Xibing Li, and Zhong Qi Yue. Structural reliability analysis for implicit performance functions using artificial neural network. *Structural Safety*, 27(1):25–48, 2005.

[42] Jorge E Hurtado. Filtered importance sampling with support vector margin: a powerful method for structural reliability analysis. *Structural Safety*, 29(1):2–15, 2007.

[43] João B Cardoso, João R de Almeida, José M Dias, and Pedro G Coelho. Structural reliability analysis using monte carlo simulation and neural networks. *Advances in Engineering Software*, 39(6):505–513, 2008.

[44] Jin Cheng, QS Li, and Ru-cheng Xiao. A new artificial neural network-based response surface method for structural reliability analysis. *Probabilistic Engineering Mechanics*, 23(1):51–63, 2008.

[45] E Volkova, B Iooss, and F Van Dorpe. Global sensitivity analysis for a numerical model of radionuclide migration from the rrc kurchatov institute radwaste disposal site. *Stochastic Environmental Research and Risk Assessment*, 22(1): 17–31, 2008.

[46] Amandine Marrel, Bertrand Iooss, Beatrice Laurent, and Olivier Roustant. Calculations of sobol indices for the gaussian process metamodel. *Reliability Engineering & System Safety*, 94(3):742–751, 2009.

[47] Piercesare Secchi, Enrico Zio, and Francesco Di Maio. Quantifying uncertainties in the estimation of safety parameters by using bootstrapped artificial neural networks. *Annals of Nuclear Energy*, 35(12):2338–2350, 2008.

[48] Uchenna Oparaji, Rong-Jiun Sheu, Mark Bankhead, Jonathan Austin, and Edoardo Patelli. Robust artificial neural network for reliability and sensitivity analyses of complex non-linear systems. *Neural Networks*, 96:80–90, 2017.

[49] Andrea GB Tettamanzi and Marco Tomassini. *Soft computing: integrating evolutionary, neural, and fuzzy systems*. Springer Science & Business Media, 2013.

[50] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[51] Paul John Werbos. Beyond regression: New tools for prediction and analysis in the behavioral sciences. *Doctoral Dissertation, Applied Mathematics, Harvard University, MA*, 1974.

[52] Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[53] James Kennedy. Particle swarm optimization. In *Encyclopedia of machine learning*, pages 760–766. Springer, 2011.

[54] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[55] Jorge J Moré. The levenberg-marquardt algorithm: implementation and theory. In *Numerical analysis*, pages 105–116. Springer, 1978.

[56] Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 2012.

[57] Anders Krogh and Jesper Vedelsby. Neural network ensembles, cross validation, and active learning. In *Advances in neural information processing systems*, pages 231–238, 1995.

[58] Robert E Schapire. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*, pages 149–171. Springer, 2003.

[59] Bradley Efron. Bootstrap methods: another look at the jackknife. In *Breakthroughs in statistics*, pages 569–593. Springer, 1992.

[60] ED Cashwell and CJ Everett. *Monte-Carlo methods*. Pergamon, London, 1959.

[61] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

[62] Marco Rigamonti, Piero Baraldi, Enrico Zio, Indranil Roychoudhury, Kai Goebel, and Scott Poll. Ensemble of optimized echo state networks for remaining useful life prediction. *Neurocomputing*, 2017.

[63] Thomas Nilsen and Terje Aven. Models and model uncertainty in the context of risk analysis. *Reliability Engineering & System Safety*, 79(3):309–317, 2003.

[64] Enrico Zio. A study of the bootstrap method for estimating the accuracy of artificial neural networks in predicting nuclear transient processes. *IEEE Transactions on Nuclear Science*, 53(3):1460–1478, 2006.

[65] MJ Bayarri, JO Berger, John Cafeo, G Garcia-Donato, F Liu, J Palomo, RJ Parthasarathy, R Paulo, Jerry Sacks, and D Walsh. Computer model validation with functional output. *The Annals of Statistics*, pages 1874–1906, 2007.

[66] Maria J Bayarri, James O Berger, Rui Paulo, Jerry Sacks, John A Cafeo, James Cavendish, Chin-Hsu Lin, and Jian Tu. A framework for validation of computer models. *Technometrics*, 2012.

[67] Marc C Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464, 2001.

[68] David G Kleinbaum and Mitchel Klein. Maximum likelihood techniques: An overview. In *Logistic regression*, pages 103–127. Springer, 2010.

[69] Thomas H Wonnacott and Ronald J Wonnacott. *Introductory statistics*, volume 19690. Wiley New York, 1972.

[70] Il Yong Kim and Oliver L de Weck. Adaptive weighted-sum method for bi-objective optimization: Pareto front generation. *Structural and multidisciplinary optimization*, 29(2):149–158, 2005.

[71] Andrew R Barron. Predicted squared error: a criterion for automatic model selection. *Unknown*, 1984.

[72] John E Moody. Note on generalization, regularization and architecture selection in nonlinear learning systems. In *Neural Networks for Signal Processing [1991]., Proceedings of the 1991 IEEE Workshop*, pages 1–10. IEEE, 1991.

[73] Weimin Dong and Haresh C Shah. Vertex method for computing functions of fuzzy variables. *Fuzzy sets and Systems*, 24(1):65–78, 1987.

[74] William V Harper and Sumant K Gupta. Sensitivity/uncertainty analysis of a borehole scenario comparing latin hypercube sampling and deterministic sensitivity approaches. Technical report, Battelle Memorial Inst., Columbus, OH (USA). Office of Nuclear Waste Isolation, 1983.

[75] CD Fletcher and RR Schultz. Relap5/mod3 code manual volume v: Users guidelines. *Idaho National Engineering Laboratory, Lockheed Idaho Technologies Company, Idaho Falls, Idaho*, 83415, 1995.

[76] Alfred Klett, Sabine Mayer, Christian Theis, and Helmut Vincke. A neutron dose rate monitor for high energies. *Radiation measurements*, 41:S279–S282, 2006.

[77] Alberto Fassò, James C Liu, and Sayed H Rokni. Neutron spectra and dosimetric quantities outside typical concrete shielding of synchrotron facilities. *ICRS-12 & RPSD-2012, Nara, Japan*, pages 2–7, 2012.

[78] Richard H Olsher, Hsiao-Hua Hsu, Anthony Beverding, Jeffrey H Kleck, William H Casson, Dennis G Vasilik, and Robert T Devine. Wendi: An improved neutron rem meter.. *Health Physics*, 79(2):170–181, 2000.

[79] L Jagerhofer, Eduard Feldbaumer, Doris Forkel-Wirth, Chris Theis, Helmut Vincke, Yosuke Iwamoto, Masayuki Hagiwara, Daiki Satoh, Hiroshi Iwase, Hiroshi Yashima, et al. Characterization of the wendi-ii rem counter for its application at medaustron. *Prog. Nucl. Sci. Technol*, 2:258–262, 2011.

[80] TSR IAEA. 403: Compendium of neutron spectra and detector responses for radiation protection purposes, 2001.

[81] DB Pelowitz. Mcpx user's manual version 2.7. 0 (los alamos: Los alamos national laboratory). Technical report, LA-CP-11-00438, 2011.

[82] H Schuhmacher. Neutron calibration facilities. *Radiation protection dosimetry*, 110(1-4):33–42, 2004.

[83] International Commission on Radiological Protection. Conversion coefficients for use in radiological protection against external radiation. *ICRP Publications*, 74: 365–382, 1996.

[84] M Pelliccioni. Overview of fluence-to-effective dose and fluence-to-ambient dose equivalent conversion coefficients for high energy radiation calculated using the fluka code. *Radiation Protection Dosimetry*, 88(4):279–297, 2000.